МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

МЕТОДИЧНІ ВКАЗІВКИ І ЗАВДАННЯ ДО КОНТРОЛЬНОЇ РОБОТИ ПО КУРСУ ПРОГРАМУВАННЯ

ЧАСТИНА 4

(для студентів спеціальності 7.091501
«Комп'ютерні системи та мережі»
заочної форми навчання)

Розглянуто
на засіданні кафедри КІ
Протокол № 1
від 31 серпня 2010 р.

Затверджено
на засіданні навчально-
видавницької ради ДонНТУ
Протокол № 5 від 06.12.10

Донецьк ДонНТУ 2010

УДК 681.3(07)

МЕТОДИЧНІ ВКАЗІВКИ І ЗАВДАННЯ ДО КОНТРОЛЬНОЇ РОБОТИ ПО КУРСУ ПРОГРАМУВАННЯ, ЧАСТИНА 4 (для студентів спеціальності 7.091501 «Комп'ютерні системи та мережі» заочної форми навчання). Складачі: В.І.Назаренко, О.Ю.Чередніков, К.Б.Юсупова. – Донецьк, ДонНТУ, 2010. – 66 с.

Приведені методичні вказівки до контрольної роботи № 4 по курсу програмування для студентів-заочників спеціальності 7.091501. Контрольна робота містить завдання по темі «Використання лінійних списків і записів у складі багатомодульної програми». Для виконання контрольної роботи наведено 50 варіантів завдань. Матеріал посібника містить також необхідний теоретичний матеріал і приклад виконання одного із завдань.

Складачі: доц.Назаренко В.І.
ас.Чередніков О.Ю.
ас. Юсупова К.Б.

Відповідальний
за випуск проф.Святний В.А.

Рецензент доц.Федяєв О.І.

З А Г А Л Ь Н І М Е Т О Д И Ч Н І В К А З І В К И

Контрольна робота № 4 для студентів-заочників спеціальності 7.091501 "Комп'ютерні системи та мережі" містить у собі завдання, орієнтовані на розробку і виконання багатомодульных програм мовою Турбо Паскаль.

Контрольна робота, представлена студентом на кафедру, повинна містити:

- умову завдання;
- короткий опис алгоритму і програми;
- лістинг програми;
- результати рішення завдання.

У методичному посібнику наведено 50 варіантів завдань. Вибір варіанта виконується по останнім двом цифрам номера залікової книжки, збільшеним на 1. Якщо отримане число перевищує значення 50, то воно зменшується на 50.

Тексти контрольної роботи, у тому числі лістинги програм, треба виконувати на стандартних аркушах паперу формату А4. При цьому на титульному аркуші повинне бути написано:

Донецький національний технічний університет
Кафедра комп'ютерної інженерії

Контрольна робота № 4
за курсом "Програмування"

ст-та гр. (шифр групи)
Прізвище і ініціали

Номер залікової книжки ...

Домашня адреса: ...

Текст програми (лістинг) і результати рішення завдання повинні бути видрукувані на принтері. Для цього доцільно використовувати безпосередньо можливості компілятора мови Турбо Паскаль. Якщо ж за якимось причинами друк тексту програми здійснюється за допомогою текстового процесора Word, то для цього потрібно використовувати шрифт Courier New розміром 12 (у шрифті Courier New кожний символ тексту займає окрему позицію). У тексті програми в обов'язковому порядку повинні бути виконані вимоги по стилю програмування (ці вимоги викладені в першій частині методичних вказівок).

Слід зазначити, що текст програми, наведеної в даних методвказівках як приклад виконання типового завдання, видрукуваний за допомогою текстового

процесора Word. Тому механічний перенос його в операційне середовище MS DOS для компіляції транслятором Turbo Pascal пов'язаний з певними труднощами. У той же час багато фрагментів цієї програми, у першу чергу сервісні процедури й функції, можуть бути використані студентами при виконанні свого завдання. У зв'язку з цим студентам рекомендується переписати на дискету в дисплейному класі кафедри КІ оригінальний текст зазначеного приклада (файли програмних модулів DesUnit.pas, BasUnit.pas, InpUnit.pas, WorkUnit.pas, Kontr4.pas, файли вхідних даних Input.txt, Kodif.txt, Add.txt і файл Readme.txt з коротким поясненням приклада). Включення зазначених фрагментів у свою програму шляхом копіювання файлу принаймні заощаджує час, нераціонально затрачуваний при наборі їх із клавіатури.

М Е Т О Д И Ч Н І В К А З І В К И

по темі контрольної роботи

Мета роботи - освоїти методи проектування багатомодульных програм, що обробляють лінійні списки, інформаційною частиною яких є записи.

1. З а г а л ь н і з а у в а ж е н н я

У широко розповсюджених адміністративно-облікових системах (АОС), які називають також автоматизованими системами управління (АСУ), об'єктом машинної обробки є документ (група документів).

АСУ з погляду програмної реалізації має ряд характерних ознак.

1. Використання великої кількості файлів для подання вхідної, корегувальної, обробної і результуючої інформації.

2. Превалювання операцій введення-виводу в загальному часі рішення завдання в порівнянні з обчислювальними операціями (на відміну від науково-технічних задач).

3. Об'єднання в одному комплексі великої кількості функціональних режимів роботи системи (уведення певної категорії даних, перевірка їхньої вірогідності, корекція даних у файлах, різні види обробки даних і т.д.). При цьому, як правило, конкретні режими роботи системи реалізуються окремими процедурами. Вибір режиму роботи виконується користувачем у процесі діалогу з ЕОМ.

Будь-який документ структурно розділяється на три частини: "шапка", колонки і рядки. Колонки документа, що відображають його змістовну частину, називають також реквізитами.

У документах, оброблюваних на ЕОМ, часто зустрічаються повторювані текстові реквізити (наприклад, найменування кафедри, назва навчальної дисципліни в АСУ вузу і т.п.). Для зменшення витрат пам'яті при зберіганні і

обробці документів такі реквізити заміняють кодами (шифрами), а відповідність між шифрами і визначальними їхніми текстами відображають в окремих файлах, які називають кодифікаторами відповідної інформації. Для вихідних документів, якщо буде потреба, по заданому шифрі автоматично визначається з кодифікатора його текстовий еквівалент, що і видається на друк.

У програмі на Паскалі кожний рядок документа - це окремий запис; документ у цілому (група однорідних документів) при обробці на ЕОМ представляється масивом або списком записів, а при тривалому зберіганні - файлом, компонентами якого є відповідні записи.

У контрольній роботі № 4 відповідно до умови завдання необхідно виконати наступне:

- уведення записів з текстового файлу й формування типізованого файлу, що розглядається надалі як архів АСУ;
- угруповання архіву по певній ознаці;
- вивід на екран і друку на принтері архіву АСУ;
- доповнення архіву новими записами;
- вилучення заданого запису з архіву;
- зміна реквізитів заданого запису;
- формування вихідних документів.

Вихідний текстовий файл, як правило, повинен містити не менш 25 записів. Доповнення архіву повинне вироблятися з використанням окремого текстового файлу, що містить 3 - 5 записів.

Використання кодифікатора при виконанні контрольної роботи необов'язково.

2. У в о д з а п и с і в

Припустимо, що особиста картка студента містить наступні реквізити: прізвище і ініціали, дата народження, національність, рік закінчення школи, стать. Цій картці поставимо у відповідність наступний опис запису:

```
Type  Date = record
         Day : 1..31;                { число }
         Month : 1..12;              { місяць }
         Year : 1970 .. 2006;        { рік }
      end;
      StudType = record
         Fam : string[25];    { прізвище і ініціали }
         BirthDay : Date;     { дата народження }
         Nac : string[15];    { національність }
         SchoolYear           { рік закінчення школи }
             : 1990..2000;
         Sex : char           { стать }
      end;
Var  Student : StudType;      { особиста картка }
```

```
F : text;        { вихідний текстовий файл }
```

У даному прикладі для оголошення компонентів Day, Month, Year, SchoolYear використаний діапазонний тип. З погляду обсягу виділюваної пам'яті ці оголошення не відрізняються від наступних:

```
Day,Month : byte;
Year,ShoolYear : word.
```

Однак використання діапазонного типу дозволяє контролювати присвоювання таким змінним нових значень (наприклад, при уведенні) і полегшує розуміння програми, явно вказуючи в її тексті на припустимі діапазони зміни відповідних даних.

Контроль присвоювання ординальным змінним нових значень здійснюється автоматично, якщо включено директиву компілятора R. При наявності такої директиви транслятор вставляє в програму групи додаткових машинних команд, що збільшує її обсяг і сповільнює виконання. Тому директиву R рекомендується використовувати лише в процесі налагодження програми.

Розглянемо два варіанти уведення запису з текстового файлу F. В обох варіантах передбачається, що реквізити запису розміщуються в одному рядку текстового файлу. Значення реквізитів розділяються між собою одним або декількома пропусками.

Варіант 1.

Кількість позицій, що відводяться в рядку файлу кожному текстовому реквізиту, дорівнює або більше значення, зазначеного в описі відповідного рядка.

Приклад рядка файлу F:

```
Петренко А.С.            15 5 1980 українець        1977  м
```

Тоді програма уведення одного запису може мати такий вигляд:

```
With Student do
  Begin
    Read(F,Fam);
    Read(F,BirthDay.Day,BirthDay.Month,
           BirthDay.Year);
    Repeat
      Read(F,ch);
    Until  ch<>' ';
    Read(F,Nac);
```

```
    Insert(ch,Nac,1);
    Read(F,SchoolYear);
    Repeat
      Read(F,ch);
    Until  ch<>' ';
    Sex:=ch;
    Readln(F);
  End;
```

Коментарі до програми:

1. У програмі передбачається, що змінна Student.Fam записана з першої позиції рядка текстового файлу. Якщо це не так, то пропуски, поставлені в рядку файла перед реквізитом "прізвище і ініціали", будуть включені до складу змінної Student.Fam, що надалі відіб'ється на коректності роботи програми.

2. В одному рядку файлу одночасно розміщені різнотипні елементи: числа й рядки. Ці елементи відділені один від другого одним або декількома пропусками. Процедура Read при уведенні числової змінної автоматично виділяє її з рядка текстового файлу. При уведенні змінної типу **string** пропуск сприймається як звичайний символ. Тому в програмі для індикації початку текстової змінної за допомогою циклу **Repeat** перебираються позиції рядка файлу доти, поки в символьну змінну ch не буде прочитаний символ, відмінний від пропуску, після чого читається строкова змінна Nac.

Оскільки перший символ реквізиту "національність" був прочитаний у символьну змінну ch, то його додавання в змінну Nac виконується процедурою Insert.

3. Аналогічно за допомогою циклу Repeat виконується індикація символу, що відповідає реквізиту Sex.

4. Процедура Readln(F) виконує перехід на новий рядок текстового файлу для уведення наступного запису.

Припустимо, що при формуванні текстового файлу не дотримана відповідність між кількістю позицій у рядку файлу й оголошенням змінної типу **string**. Наприклад, для змінної Fam відведено менше 25 позицій:

```
Петренко А.С.        15 5 1980  українець        1977  м
```

Тоді введене значення змінної Fam у цьому випадку буде мати вигляд:

```
'Петренко А.С.        15 5',
```

для змінної BirthDay.Day буде уведене значення 1980 (насправді ця змінна, що відповідає типу byte, одержить значення 188 - останній байт значення 1980 у форматі word (07BC)), а при спробі привласнити змінній BirthDay.Month значення "українець" відбудеться переривання програми з виводом повідомлення "Runtime Error 106: Invalid numeric format" (“Помилка періоду виконання 106: Неправильний числовий формат”).

З погляду користувача, варіант 1 має серйозний недолік: при підготовці текстового файлу необхідно постійно стежити, щоб значення змінної Student.Fam займало в рядку файлу не менше 25 позицій, а змінної Student.Nac - не менше 15 позицій, поза залежністю від поточної довжини цих змінних. У той же час для підвищення сервісу користувача більш зручно було б залишити лише одне обмеження: елементи рядка файлу повинні розділятися одним або декількома пропусками. Приклад реалізації уведення записів Student з текстового файлу з урахуванням зазначеного обмеження наведений нижче як варіант 2.

Варіант 2.

Кількість позицій, що відводяться в рядку файлу кожному текстовому реквізиту, визначається лише його поточною довжиною.

Приклад рядка файлу F:

```
Петренко А.С.  15 5 1980  українець  1977  м
```

Програма уведення одного запису може мати вигляд:

```
Var   k,k1,k2 : byte;
      Code : integer;
      Cond : boolean;
      Sa,Sb : string;
  With Student do
    Begin
      Readln(F,Sa);
      k:=0; k2:=0;
      Cond:=true;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1;
                  Cond:=false;
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Fam:=Sb;
```

```
                2 : Begin
                      Fam:=Fam+' '+Sb;
                      While length(Fam)<25 do
                        Fam:=Fam+' ';
                    End;
                3 : Val(Sb,BirthDay.Day,Code);
                4 : Val(Sb,BirthDay.Month,Code);
                5 : Val(Sb,BirthDay.Year,Code);
                6 : Begin
                      Nac:=Sb;
                      While length(Nac)<15 do
                        Nac:=Nac+' ';
                    End;
                7 : Val(S2,ScoolYear,Code);
                8 : Sex:=Sb[1];
              end;
            End;
        End;
    End;
```

У варіанті 2 черговий рядок текстового файлу вводиться в строкову змінну Sa. Після цього за допомогою функцій Space і NotSpace, розглянутих у контрольній роботі № 3, виділяються окремі слова рядка Sa, формування компонентів запису Student виконується оператором **Case** відповідно до порядкового номера k виділеного слова. При цьому враховано, що між прізвищем і ініціалами студента в рядку вхідного файлу є принаймні один пропуск.

Строкові змінні Fam і Nac оператором **While** доповнюються пропусками до оголошеної довжини. Вимога чітко витримувати оголошену довжину строкової змінної обумовлена тим, що в програмі надалі може виконуватися пошук запису на прізвище студента або зміна якого-небудь текстового реквізиту, використовуючи при цьому порівняння строкової змінної зі значенням, що вводиться із клавіатури.

Варто звернути увагу на одну помилку, що допускається часто при формуванні текстового файлу.

Якщо наприкінці текстового файлу є один або кілька рядків, заповнених пропусками, то програмою буде уведена відповідна кількість додаткових записів. У варіанті 1 ці записи будуть заповнені нульовими значеннями, у варіанті 2 вони будуть копіювати останный "нормальний" запис. Щоб виключити порожні рядки наприкінці текстового файлу, рекомендується наступне:

1. У середовищі редактора Turbo Pascal або Norton Commander (клавіша F4 - Edit) нажати клавіші Ctrl + PgDn, внаслідок чого курсор буде переміщений у кінець файлу.

2. Якщо курсор установиться нижче останнього значущого рядка текстового файлу, то примусово установити його після останнього символу останнього рядка файлу ы виконати тривале натискання клавіші Del (3-5 с).

3. Перевірити відсутність порожніх рядків послідовним натисканням клавішей Ctrl+PgUp, а потім Ctrl+PgDn (перехід курсору на початок, а потім у кінець файлу).

Неважко помітити, що ці рекомендації адресовані користувачеві, а не розроблювачеві, і оскільки їхнє дотримання затруднює роботу користувача, то програміст повинен прийняти такі міри, при яких програма буде працювати правильно поза залежністю від наявності або відсутності порожніх рядків у текстовому файлі. Цього можна досягти принаймні двома методами:

1) порожній рядок, що читається з текстового файлу, не сприймати як уміст чергового запису;

2) попередньо програмно перевірити текстовий файл і вилучити з нього порожні рядки, якщо вони є.

У прикладі виконання контрольної роботи (програма Kontr4) реалізовано другий метод рішення зазначеної проблеми.

3. Використання лінійних списків

Будемо розглядати наведений вище приклад по обробці особистих карток студентів.

У ряді режимів обробки потрібна наявність у пам'яті ЕОМ одночасно всіх особистих карток. Така необхідність виникає, наприклад, при угрупованні карток за алфавітом прізвищ студентів.

Рішення розглянутого завдання може бути виконане шляхом використання масиву або лінійного списку записів.

Доповнимо опис запису наступним фрагментом:

```
Const  Nmax = 500;  { максимальний розмір масиву Studs }
Type  StudAr = array[1..Nmax] of StudType;
Var    n : word;    { поточний розмір масиву Studs }
   Studs : StudAr;  { масив особистих карток }
```

Тут оброблюваний архів документів представлений в пам'яті ЕОМ у вигляді масиву Studs, кожний компонент якого - це запис типу StudType, що має довжину 49 байт.

З погляду машинної обробки використання масиву для подання групи оброблюваних записів, що зберігаються в архівному файлі, має два істотних недоліки.

1. Масив завжди має фіксований розмір (у цьому випадку Nmax = 500), який не може бути змінений у процесі роботи програми. Отже, при оголошенні

масиву доводиться орієнтуватися на його максимально можливий розмір, що веде до неощадливого використання пам'яті ЕОМ.

2. При зміні поточної кількості компонентів у масиві (додавання нових або вилучення існуючих) оброблюваний масив потрібно повністю або частково перемістити в пам'яті ЕОМ (зрушувати вліво або вправо компоненти масиву). Це вимагає певних витрат машинного часу, які тим більше, чим більш довгим є поле пам'яті, яке займає один компонент.

Зазначені недоліки можна усунути, якщо для подання групи оброблюваних документів використовувати спискові структури (стек, черга і т.п.). Для списку може бути виділено рівно стільки пам'яті, скільки потрібно для розміщення даної групи записів, причому ця пам'ять може бути виділена в будь-який момент роботи програми і звільнена, коли в цьому виникне необхідність. При зміні кількості компонентів у списку немає необхідності переміщати їх у пам'яті; для цього досить змінити значення відповідних покажчиків.

Проте спискові структури мають також деякі недоліки порівняно з масивами.

1. Компонент однонаправленого списку займає на 4 байти більше пам'яті, ніж компонент масиву (за рахунок розміщення в ньому покажчика).

2. Список не надає можливість прямого доступу до його компонентів (наприклад, читання компонента по його номері в списку). Це, зокрема, виключає можливість застосування методу двійкового пошуку для знаходження в згрупованому списку компонента із заданим кодом.

Порівнюючи достоїнства і недоліки спискових структур, можна зробити висновок, що для завдань АСУ, у яких об'єктами обробки в основному є записи великого розміру, списки більш кращі в порівнянні з масивами.

При використанні лінійного списку для подання архіву записів необхідно вибрати один із двох простих типів списку: стек або черга.

Припустимо, що записи, що читаються з текстового файлу, "складуються" у стек. Тоді відносний порядок записів, що читаються зі стека, буде зворотним стосовно послідовності записів у файлі. Це створює певні незручності при програмній обробці списку, зокрема, вимагає виконувати реверс компонентів стека. У черзі відносний порядок компонентів не відрізняється від послідовності записів у файлі. У зв'язку з цим у завданнях обробки архіву записів черга більш краща в порівнянні зі стеком. Тоді опис компонента черги може мати такий вигляд:

```
Type  PointStud = ^DynStud;
      DynStud = record
         Inf : StudType;
        Next : PointStud;
      end;
Var  LeftStud,RightStud,RunStud : PointStud;
```

Тут Inf - інформаційна частина компонента черги;

Next - покажчик на наступний елемент;

LeftStud, RightStud, RunStud - відповідно лівий, правий і поточний покажчики черги.

4. Контроль вхідних даних

Уведення вхідних даних виконується з текстового файлу, який попередньо формується із клавіатури. При цьому можливі різні помилки, внесені користувачем при заповненні цього файлу.

Такими помилками можуть бути:

- невірне ім'я файлу;
- форматна помилка в числовому реквізиті (буква замість цифри, кома замість поділяючої крапки і т.п.);
- значення числового реквізиту поза припустимим діапазоном;
- довжина рядка текстового реквізиту перевищує оголошену;
- наявність порожнього рядка та ін.

Припустимо, що в текстовому файлі при уведенні даних виявлена помилка формату. Наслідком цього є переривання роботи програми з виводом повідомлення "Runtime error 106: Invalid numeric format (Неправильний числовий формат)". Таке повідомлення для користувача, що не є програмістом, незрозуміло, а у випадку, коли відомо зміст повідомлення, залишається невизначеним, де саме виявлена помилка.

Помилки, що не викликають переривання програми (наприклад, значення числа поза припустимим діапазоном), як правило, приводять до одержання невірних результатів роботи програми. У цьому випадку для локалізації помилки потрібна від користувача кропітка робота з візуальної перевірки набору вхідних даних.

Обов'язковим компонентом реальних програм є автоматична перевірка коректності вхідних даних з формуванням протоколу контролю, у який записуються повідомлення про помилки (їхній тип і місце розташування). У програмі повинен бути передбачений також контроль відповідей користувача, які вводяться із клавіатури при запитах програми.

Можлива методика контролю коректності даних реалізована в прикладі виконання контрольної роботи (процедури CheckFiles модуля InpUnit і GetNumber модуля BasUnit).

5. Розробка багатомодульної програми

Паскаль-програма не може займати понад 64 Кбайт оперативної пам'яті. Тому при розробці великих програм використовують апарат модулів користувача. У цьому випадку програма розділяється на кілька програмних

модулів. Кожний модуль також не може мати розмір понад 64 Кбайт, але кількість модулів у програмі не обмежується. Важливою властивістю модуля є також те, що його можна компілювати й налагоджувати автономно, незалежно від інших модулів. Це прискорює процес створення великої програми.

Модуль розділяється на три частини: секція інтерфейсу, секція реалізації і секція ініціалізації.

Заголовок модуля складається із зарезервованого слова **Unit** і наступного за ним ідентифікатора, що є ім'ям модуля. Ім'я модуля повинне бути унікальним. Передбачається, що ім'я модуля збігається з ім'ям файлу, у якому зберігається даний модуль. Наприклад, модуль із заголовком **Unit** `BasUnit` повинен перебувати у файлі з ім'ям "BasUnit.pas". Наслідком останньої обставини є те, що ім'я модуля, як і ім'я файлу, повинне складатися не більш ніж з восьми символів.

Інтерфейсна частина починається зі службового слова **Interface.** Якщо в даному модулі використовуються інші модулі (стандартні або модулі користувача), то після слова **Interface** повинна бути записана фраза **Uses** з іменами використовуваних модулів. Інтерфейсна частина може містити описи констант, типів, змінних, процедур і функцій. Ці описи вважаються глобальними, їх може використовувати будь-який інший модуль, у фразі **Uses** якого зазначене ім'я даного модуля.

Unit UnitName	
Interface	
----------i	Секція
Опис видимих об'єктів	інтерфейсу

Implementation	
----------	Секція
Опис схованих об'єктів	реалізації

Begin	
----------	Секція
Оператори ініціалізації	ініціалізації

End.	

Як відомо, заголовок підпрограми містить всю інформацію, необхідну для її виклику: ім'я підпрограми, кількість і типи параметрів і, якщо це функція, тип результату. Тіло підпрограми - це блок, що визначає алгоритм її роботи. З погляду визиваючої програми вся необхідна й достатня інформація

знаходиться в заголовку підпрограми, блок підпрограми носить сугубо внутрішній характер стосовно викликуваної підпрограми.

У зв'язку з вищесказаним в інтерфейсній частині розміщають тільки заголовки процедур і функцій, що мають значення при їхньому глобальному використанні. Повний опис процедур і функцій переноситься в секцію реалізації, що починається зі службового слова **Implementation.**

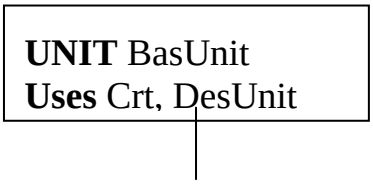
У секції реалізації можуть бути свої описи констант, типів і змінних, використовуваних тільки при роботі цього модуля. Тут можуть міститися також процедури і функції, заголовки яких відсутні в секції інтерфейсу; ці процедури і функції недоступні для інших модулів і можуть бути активізовані тільки при роботі даного модуля.

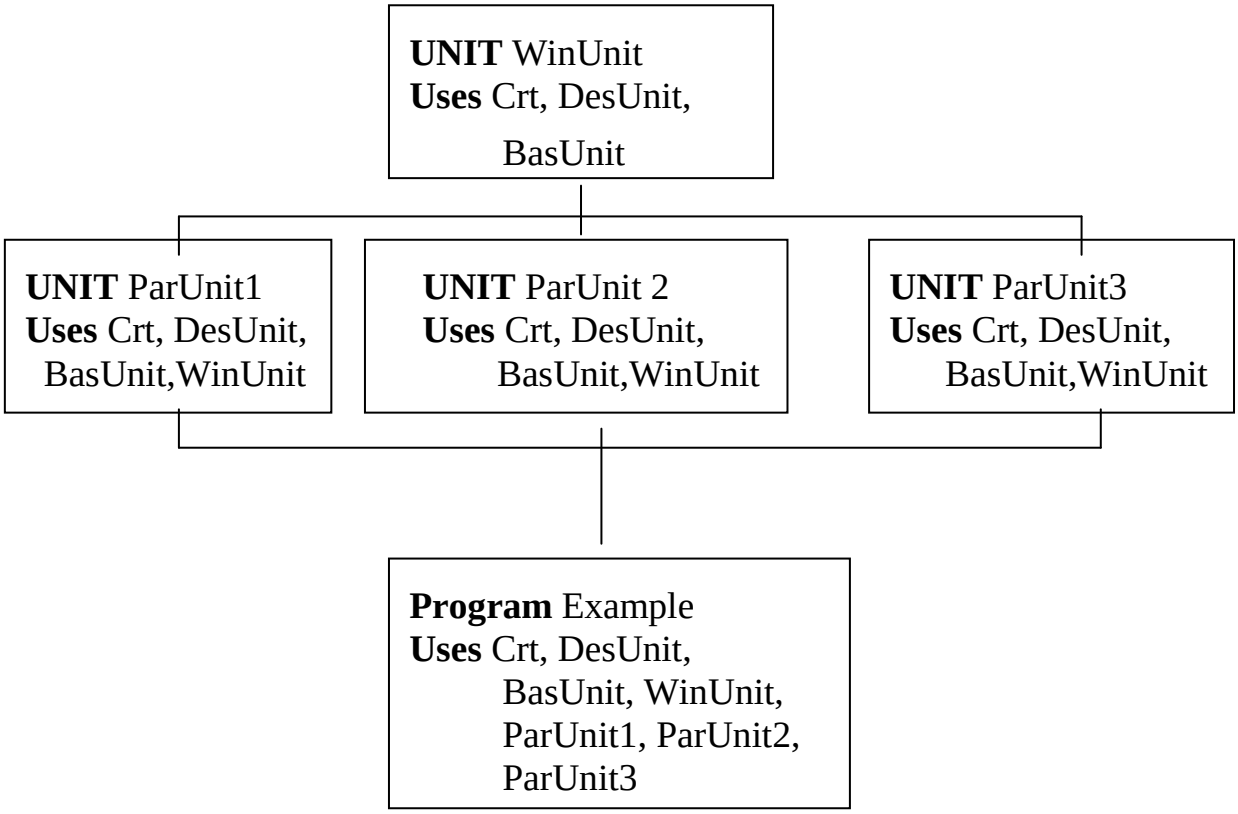
Оскільки інші модулі можуть використовувати тільки об'єкти, описані в секції інтерфейсу, але не в секції реалізації, то говорять, що секція інтерфейсу містить видимі об'єкти, а секція реалізації - сховані об'єкти. Розходження між секціями інтерфейсу і реалізації можна визначити ще в такий спосіб: секція інтерфейсу вказує, що робить модуль, а секція реалізації описує, як він це робить.

Секція ініціалізації є необов'язкової, її наявність визначає слово **Begin.** Секції ініціалізації всіх модулів викликаються перед запуском основного тіла програми. Ці секції звичайно використовують для установки початкових значень змінних, для відкриття файлів, автовизначення типу монітора та ін. Секція ініціалізації може бути порожньою, а при відсутності слова **Begin** відповідний модуль не містить цю секцію.

Структура багатомодульної програми може бути всілякою залежно від призначення програми, її розміру, об'єднання процедур по групах і т.д. Нижче приводиться один з можливих варіантів такої структури.

Тут у модулі DesUnit розміщені описи глобальних констант, типів і змінних (Des - від слова description, опис). Секція реалізації цього модуля порожня. У модулі BasUnit зосереджені базові службові процедури й функції, що використовуються в багатьох інших модулях (функція знака, процедури контролю наявності файлів, двійкового пошуку, угруповання масивів, виводу масиву на екран, вилучення черги і т.п.). У модулі WinUnit записані процедури для організації багатовіконного інтерфейсу (включення і відключення курсору, збереження і відновлення екрана, формування різного типу меню і т.п.). Модулі ParUnit1, ParUnit2, ParUnit3 містять процедури і функції, що реалізують різні режими роботи програми. В основній програмі Example виконується вибір режиму її роботи шляхом активізації позицій меню і звертання до відповідних процедур.





Компіляція модуля можлива як з інтегрованого середовища Турбо Паскаля (turbo.exe), так і за допомогою компілятора командного рядка (tpc.exe). Оскільки модуль не є безпосередньо виконуваною одиницею, то в результаті його компіляції утвориться дисковий файл із розширенням tpu (turbo pascal unit), при цьому ім'я такого файлу копіюється з ім'я файлу з вхідним текстом модуля. Компіляція модуля можлива, якщо до початку компіляції створені всі нестандартні модулі з розширенням tpu, імена яких перераховані у фразі **`Uses`** даного модуля.

6. Приклад виконання завдання

Умова завдання.

Задані наступні відомості про діяльність цехів деякого підприємства: номер цеху (2 цифри), код виробу (6 цифр), найменування виробу, одиниця виміру, ціна виробу, план випуску і фактичний випуск виробу по півріччях. Необхідно створити архів відомостей про вироби, що випускаються, і згрупувати документи архіву по зростанню коду виробу. У програмі повинні бути передбачені наступні види корекції архіву:

- додавання документів;
- вилучення документа;
- зміна в заданому документі останніх чотирьох реквізитів (одиниця виміру, ціна, плановий і фактичний випуски).

В обробній частині програми передбачити:

- друк архіву;
- визначення для заданого цеху відсотка виконання плану у вартісному вираженні по півріччям і в цілому за рік;
- формування списку виробів, по яким не виконаний річний план виробництва.

Реалізація завдання наведена в описуваній нижче програмі Kontr4.

О п и с п р о г р а м и K o n t r 4

Програма Kontr4 складається з п'яти модулів:

- DesUnit - глобальні описи констант, типів і змінних;
- BasUnit - сервісні процедури і функції;
- InpUnit - уведення і контроль коректності текстових файлів;
- WorkUnit - обробні процедури і функції;
- Kontr4 - основна програма.

Рядку документа, наведеного в умові завдання, у програмі (модуль DesUnit) відповідає запис типу ProductType. Деяких пояснень вимагає подання в даному записі двох реквізитів: номера цеху NumberShop і коду виробу KodProduct.

В умові зазначено, що номер цеху зображується двома цифрами. Із цього погляду даний параметр можна трактувати як число. Однак це число має специфічну особливість: воно не може бути використане для арифметичних операцій. Безглуздо, наприклад, говорити про суму двох номерів цехів або про їхній добуток. Номер цеху при обробці в програмі може використовуватися лише в операціях порівняння (наприклад, при угрупованні записів по зростанню номера цеху). Оскільки арифметичні операції для номера цеху NumberShop неприпустимі, те змінну NumberShop можна було б представити як рядок типу **`string`**`[2]`, елементами якої є цифри. У програмі для змінної NumberShop обраний тип byte в основному з погляду економії пам'яті (тип byte вимагає один байт пам'яті замість трьох байтів для типу string[2]).

З аналогічних міркувань для змінної KodProduct прийнятий тип longint (максимальне значення змінної KodProduct за умовою завдання дорівнює 999999; максимальне значення для типу byte дорівнює 255, для типу integer - 32767, типу word - 65535, типу longint - 2 147 483 647). Якби за умовою завдання для коду виробу використовувався буквено-цифровий шифр або ж цифровий код займав більше 9 цифр, то змінна KodProduct повинна була б мати подання у вигляді рядка.

Відповідність між кодом виробу і його найменуванням відображено в кодифікаторі виробів, запис якого має тип KodifType. Наявність такого кодифікатора дозволяє при наборі вхідних даних не вказувати в документі найменування виробу, що прискорює підготовку даних і зменшує кількість можливих помилок. Використання кодифікатора, як було вже відзначено вище,

дозволяє також заощаджувати пам'ять при зберіганні й обробці інформації на ЕОМ.

Кількість збережених і оброблюваних документів в АСУ є змінною величиною. У процесі експлуатації АСУ типовими операціями є додавання нових документів до наявного архіву, вилучення частини документів, зміна змісту деяких документів. Кодифікатори, на відміну від основних файлів АСУ, змінюються дуже рідко; у зв'язку із цим у програмі не передбачені операції по коректуванню кодифікатора. Максимальний розмір кодифікатора визначений константою MaxKodif, його реальний розмір - змінної nk. Архівний масив реалізується у вигляді черги записів, компоненти якої мають тип PointerProduct.

У програмі Kontr4 використовуються 6 файлів. У текстовому файлі FileInput утримуються вхідні документи (відомості про вироби), у файлі FileKodif - кодифікатор виробів. Текстовий файл FileAdd призначений для документів, що додаються в архів; рядки цього файлу мають такий же вигляд, як і рядки файлу FileInput. Файл FileOut є типізованим, його компоненти мають тип записів ProductType. У текстовий файл FileRes можуть бути записані за вказівкою користувача результати обробки архіву. У файл FileError, оголошений локально в модулі InpUnit, виконується запис протоколу перевірки коректності вхідних даних.

Інформація для текстових файлів підготовлюється вручну із клавіатури ПЕОМ, що заздалегідь припускає наявність у ній різного роду помилок. Уміст текстового файлу легко змінити за допомогою будь-якого текстового редактора, що створює небезпеку внесення у вхідну інформацію навмисних або ненавмисних перекручувань. Тому використовувати текстові файли як безпосереднє джерело оброблюваних даних в АСУ недоцільно.

У реальних системах вхідні документи, що містяться в текстових файлах, ретельно контролюються в програмі з метою виявлення можливих помилок. Після виправлення виявлених помилок уміст текстового файлу видається на друк для візуального контролю; ретельно перевірені документи програмно пересилаються в типізовані файли, що представляють собою архів системи.

З погляду машинної обробки типізовані файли мають ряд переваг у порівнянні з текстовими файлами:

- можливість прямого доступу;
- більш ощадливе розміщення даних у пам'яті;
- більш швидке виконання операцій вводу-виводу.

У текстовому файлі інформація має літерне подання (кожний символ тексту або цифра числа займають окремий байт). У типізованому файлі подання інформації не відрізняється від внутрішньомашинного. Тому при звертанні до текстового файлу затрачається значний машинний час на перетворення інформації із зовнішнього подання у внутрішнє (або навпаки).

У програмі Kontr4 для зберігання архіву передбачений файл FileOut. При переписі інформації з файлу FileInput у файл FileOut попередньо виконується контроль коректності вхідних даних. Контролю піддається також уміст

текстових файлів FileKodif і FileAdd. Результати обробки архіву видаються на екран, а також за вказівкою користувача можуть друкуватися на принтері і записуватися у вихідний текстовий файл FileRes.

У програмі Kontr4 передбачені наступні режими роботи:

- контроль текстових файлів;
- створення архіву виробів;
- сортування компонентів архіву;
- друк архіву виробів;
- друк кодифікатора виробів;
- додавання компонентів в архів;
- вилучення компонента з архіву;
- зміна компонента в архіві;
- обробка архіву.

Вибір режимів роботи реалізований в основній частині програми. Спочатку на екран видається запит про режим; після уведення із клавіатури відповіді (0..9) оператор Case виконує звертання до процедури, що визначає відповідний режим.

При уведенні числової відповіді із клавіатури існує певна ймовірність помилкового набору (наприклад, буква замість цифри). При уведенні відповіді за допомогою оператора Read це викликало б переривання програми і, як наслідок, необхідність її повторного запуску.

Щоб блокувати таке переривання, уведення змінної KeyRegime проводиться за допомогою процедури GetNumber, що виконує контроль формату числа, яке вводиться, і припустимого діапазону його подання.

При помилці уведення програма видає відповідне повідомлення й пропонує користувачеві повторити уведення.

Запит режиму роботи повторюється оператором Repeat доти, поки не буде отримана відповідь KeyRegime=0. Це дає можливість користувачеві будь-яку кількість разів і в будь-якому порядку задавати необхідні режими роботи програми.

Розглянемо тепер окремо кожний з режимів роботи.

1. Контроль текстових файлів (процедура CheckFiles).

У розглянутому завданні використовуються три текстових файли: FileInput, FileAdd і FileKodif. Структура записів перших двох файлів однакова, їхня перевірка здійснюється тими самими процедурами. Для контролю параметрів третього файлу застосовуються окремі процедури.

Для повідомлень про помилки використовується типізований файл FileError, у який послідовно заноситься інформація про виявлені помилки. Індикатором наявності помилок є булівська змінна FatalError. Якщо FatalError = true, то як останній рядок у файл FileError заноситься фраза "Скорегуйте вхідні файли", у противному випадку - фраза "У вхідних файлах помилок не виявлено". У повідомленнях про помилки вказується ім'я файлу, що перевіряється, тип помилки і номер рядка, що містить помилку. Читання вмісту

файлу FileError робить процедура ReadFileError; при цьому попередньо здійснюється збереження екрана, а потім його відновлення (процедури MoveFromScreen і MoveToScreen модуля BasUnit). У кінцевій стадії роботи режиму контролю файл FileError знищується.

Помилки в текстових файлах розділені на три групи:

- помилки формату;
- помилки діапазону;
- змістовні помилки.

П о м и л к и ф о р м а т у.

Перевірка здійснюється процедурами FormatFileProduct і FormatFileKodif.

У вхідному текстовому файлі можуть бути порожні рядки або рядки, що містять тільки пропуски. Наявність таких рядків недоцільно відносити до критичних помилок, оскільки ці рядки легко усунути програмним шляхом. Для цього вміст текстового файлу вводиться в динамічний масив рядків Sf, із цього масиву вилучуються рядки, що не містять значущих символів, після чого масив Sf записується в той же текстовий файл. Ця робота виконується процедурою ReadAndCheckSpaces, звертання до якої виконується при старті процедур FormatFileProduct і FormatFileKodif.

У процедурі ReadAndCheckSpaces спочатку перевіряється наявність текстового файлу. Відсутність файлу, природно, блокує подальшу роботу з його перевірки.

Якщо текстовий файл є, то виконується читання його рядків у масив Sf. При цьому перевіряється, чи не є файл порожнім (кількість рядків nf = 0), а також, чи не перевищує його розмір максимально припустимий (nf > Nmax). У тому і іншому випадку подальша перевірка файлу не виконується.

Після успішного завершення роботи процедури ReadAndCheckSpaces відбувається безпосередня перевірка форматів параметрів. У процедурі FormatFileProduct перевіряється:

- кількість параметрів у рядку файлу (менше або більше 8);
- довжина рядка текстового параметра;
- коректність форматів числових параметрів (можливі помилки: буква замість цифри, крапка в цілочисельеному параметрі та ін.).

Процедура FormatFileKodif працює аналогічно, але з урахуванням структури запису кодифікатора.

При відсутності форматних помилок будуть сформовані динамічні масиви Products^ і AddProducts^, а також статичний масив KodifProducts.

Якщо виявлено хоча б одну помилку формату, подальший контроль текстових файлів припиняється.

П о м и л к и д і а п а з о н у.

Перевірку припустимих діапазонів числових параметрів здійснюють процедури CheckProdDiapason і CheckKodifDiapason. Тут у циклі проглядаються масиви Products^, AddProducts^ і KodifProducts. Якщо який-

небудь числовий параметр менше заданого мінімального значення або більше заданого максимального значення, то формується повідомлення про помилку.

Якщо виявлено хоча б одна помилка діапазону, подальший контроль текстових файлів припиняється.

З м і с т о в н і п о м и л к и.

Перевірка цього типу помилок виконується процедурами ProdParameters (масиви Products^ і AddProducts^) і KodifParameters (масив KodifProducts). У першій з них перевіряється:

- чи має місце дублювання коду виробу KodProduct;
- чи задане значення KodProduct у кодифікаторі;
- чи відповідає одиниця розмірності списку припустимих значень.

У процедурі KodifParameters перевіряється лише дублювання коду виробу в кодифікаторі.

2. Створення архіву виробів (процедура CreatArchive).

У процедурі CreateArchive виконується читання рядків файлу FileInput, формування запису Product (за допомогою процедури ReadProduct) і передача цього запису в типізований файл FileOut. Тут варто звернути увагу на те, що з текстового файлу читається окремо кожний компонент запису Product, а в типізований файл передається цілком весь цей запис.

3. Сортування компонентів архіву (процедура SortArchive);

На початку роботи процедури SortArchive виробляється формування черги при читанні файлу FileOut у процедурі CreateDynProduct. Для угруповання компонентів по зростанню реквізиту KodProduct застосовується метод прямої вибірки. Однак на відміну від угруповання масиву замість індексу максимального елемента при перегляді черги запам'ятовується покажчик такого елемента Pmax.

4. Друк архіву виробів (процедура PrintArchive).

Робота процедури PrintArchive цілком очевидна з її тексту. Деякі зауваження варто висловити лише із приводу використання процедури Str.

Припустимо, що нам потрібно видати на екран і на принтер (при IndPrinter = true) значення дійсних змінних *x* і *y* у форматі 8:2. Тоді можна написати

```
Writeln('x=',x:8:2,'   y=',y:8:2);
If IndPrinter then
  Writeln(Lst,'x=',x:8:2,'   y=',y:8:2);
```

Процедура Writeln завжди перетворює чисельні значення в строкові (одна цифра - один байт), після чого передає загальний рядок на вихідний пристрій. У даному прикладі змінні *x* і *y* будуть двічі перетворені із числових у строкові значення – для передачі на екран і на принтер. У процедурі PrintArchive використовується більш ефективна методика. Тут перетворення за допомогою процедури Str виконується лише один раз, після чого сумарний рядок

процедурою WritelnString (модуль BasUnit) передається на відповідний вихідний пристрій.

Вибір пристрою виводу виконується по запиту програми при її старті (процедура UsesDevice, що викликпється в початковій стадії роботи основної програми). При цьому передбачені три альтернативи виводу:

- тільки екран;
- екран і магнітний диск;
- екран і принтер.

5. Друк кодифікатора виробів (процедура PrintKodif).

У даному режимі виконується читання текстового файлу FileKodif, формування запису KodifProduct і видача її на екран і на принтер.

6. Додавання компонентів в архів (процедура AddArchive).

На завершальному етапі роботи процедури SortArchive елементи черги, згруповані по зростанню реквізиту KodProduct, пересилаються процедурою WriteFileProduct у типізований файл FileOut і, природно, зберігають у цьому файлі впорядкованість по зростанню. При старті процедури AddArchive компоненти файлу FileOut пересилаються в архівну чергу процедурою CreateDynProduct.

У процесі роботи процедури AddArchive послідовно читаються записи з файлу FileAdd. Включення цих записів у чергу виконується внутрішньою процедурою Addition.

У початковій частині процедури Addition для локального покажчика PAdd виділяється поле пам'яті процедурою New. В інформаційну частину цього поля пам'яті заноситься додається запис Product. Якщо код виробу запису, що додається, менше коду виробу першого компонента черги (Lp^.Inf.KodProduct), те поле, яке адресується покажчиком PAdd, вставляється в початок черги; у противному випадку виконується перегляд у циклі компонентів черги, що залишилися. Якщо Product.Kod більше коду виробу останнього компонента черги (Rp^.Inf.KodProduct), то новий запис додається до кінця черги; при цьому відповідно переноситься правий покажчик Rp. У противному випадку в циклі While проглядаються всі елементи черги, починаючи із другого, до виявлення елемента, код якого більше значення Product.Kod.

7. Вилучення компонента з архіву (процедура DeleteArchive).

У процедурі DeleteArchive виконується послідовний перегляд компонентів черги. Якщо при цьому буде виявлено, що реквізит KodProduct чергового компонента черги дорівнює значенню Kod, заданому із клавіатури ПЕОМ, цей компонент процедурою Dispose вилучається з пам'яті, а покажчик попереднього компонента Run^.Next приймає значення адреси Run^.Next^.Next компонента, наступного за тим, що виключається. При цьому враховані випадки, коли вилученню підлягає перший або останній елементи черги, що пов'язане зі зміною значень лівого Lp або правого Rp покажчиків.

8. Зміна компонента в архіві (процедура ChangeArchive).

Якщо при послідовному перегляді компонентів черги виявиться, що реквізит KodProduct чергового компонента дорівнює уведеному значенню Kod, то реквізитам даного компоненту, що замінюються, привласнюються значення, які вводяться із клавіатури.

9. Обробка архіву (процедура WorkUpArchive).

У даному режимі двічі проглядається архівна черга виробів. У першому випадку по заданому користувачем номеру цеху Shop вибираються із черги відповідні компоненти і підсумуються планові та фактичні показники. У результаті першого перегляду друкується таблиця відомостей про виконання плану по цеху Shop. У другому перегляді в таблицю видається список виробів, по яким не виконаний річний план виробництва. Найменування виробу вибирається з кодифікатора (масив KodifProducts), для пошуку компонента в масиві KodifProducts застосовується метод двійкового пошуку (процедура SearchKodif).

10. Уведення реквізиту із клавіатури.

У режимах вилучення і зміни компоненту користувач по запиту програми вводить із клавіатури реквізит, по якому повинен здійснюватися пошук відповідного запису в архівній черзі.

Реквізити можуть бути числові і рядкові. Уведення числового реквізиту виконується процедурами Read і Readln звичайним чином. Прикладом може служити уведення коду виробу в процедурах DeleteArchive і ChangeArchive.

Деяку особливість має уведення текстового реквізиту (або в загальному випадку рядка). Тут його виконання залежить від того, чи була безпосередньо перед цим уведенням використане процедура Read або процедура Readln.

Процедура Read очищає буфер уведення від уведеної змінної, але не переводить покажчик буфера на його початок. Процедура Readln повністю очищає буфер уведення й переводить його покажчик на початок буфера.

Нехай ми маємо:

```
Var  k : integer;
     S : string;
..............
  Read(k);
  Readln(S);
```

Оскільки після спрацьовування процедури Read покажчик буфера уведення перебував не в його початку, то рядок S одержить порожнє значення поза залежністю від того, що було набрано на клавіатурі. Така ситуація могла б виникнути в програмі Kontr4 при виконанні процедури ChangeArchive, якби в процедурі GetNumber було записано Read замість Readln.

У зв'язку з вищесказаним рекомендується при уведенні рядків завжди використовувати процедуру Readln. Якщо ж за допомогою процедури Read

уводиться одна або більше числових змінних, то після такого уведення треба записувати процедуру Readln без параметрів.

11. Використання змінної SignArchive.

У програмі Kontr4 передбачено 9 режимів роботи, вибір яких здійснює змінна KeyRegime. Звичайно, користувач може активізувати їх у будь-якому порядку, проте режими 3..9 можуть виконуватися лише після створення архіву виробів. Ця обставина враховується змінною SignArchive, що приймає значення true лише при створенні архівного файлу. Якщо SignArchive = false, то активізація інших режимів блокується з видачею на екран відповідного повідомлення.

12. Використання процедури FillString.

Припустимо, що в текстовому масиві є рядок

```
St[i] = 'abcdefgh';
```

По запиті програми користувачем було зазначено знайти індекс цього рядка в масиві. При цьому було уведено для порівняння значення

```
S = 'abcdefgh ' (3 пропуска після символу 'h').
```

При порівнянні двох рядків неоднакової довжини більш короткий доповнюється символами #0. Оскільки символ пропуску має номер #32, то програма буде вважати, що рядки St[i] і S неоднакові, а це приводить до помилок у роботі програми.

Щоб цього не трапилося, за допомогою процедури FillString кожний з уведених рядків доповнюється пропусками до оголошеної довжини.

7. Текст програми Kontr4

```
UNIT DesUnit;
{ Глобальні описи констант, типів і змінних }

Interface

Uses Crt;

Const
    MaxKodif = 50;  { макс. кількість компонентів кодифікатора }
    Enter = 13;          { код клавіші "Enter" }
    PressKey = 'Натисніть клавішу "Enter"';
Type
    ProductType = record        { тип компонента архіву виробів }
      NumberShop : byte;        { номер цеху }
             Kod : longint;     { код виробу }
          Dimens : string[5];   { одиниця виміру }
           Price : real;        { ціна виробу }
            Plan,               { план випуску по півріччях }
            Fact                { факт.випуск по півріччях }
                 : array[1..2] of real
```

```
    end;
    PointerProduct = ^DynProduct;
    DynProduct = record
         Inf : ProductType;
        Next : PointerProduct;
    end;
    KodifType = record        { тип компонента кодиф-ра виробів }
      Kod  : longint;         { код виробу }
      Name : string[35];      { найменування виробу }
    end;
    KodifAr = array[1..MaxKodif] of KodifType;

    string80 = string[80];
    StringAr = array[1..10] of string80;

Var
    np : word;                { кількість компонентів архіву }
    nk,                       { кількість комп-тів код-ра }
    KeyRegime,                { ключ вибору режиму роботи }
    Device : byte;            { пристрій виводу результатів: }
                              { 0 - екран; 1 - екран і магн.диск; }
                              { 2 - екран і принтер }
    SignArchive : boolean;    { ознака створення архіву }
    Reply : char;           { символ відповіді на запит програми }
    Product : ProductType;    { компонент архіву }
    Lp,Rp,                    { лівий і правий покажчики черги }
    Run : PointerProduct;     { поточний покажчик черги }
    Kodif : KodifType;        { компонент кодифікатора }
    Kodifs : KodifAr;         { масив компонентів кодифікатора }
    St : StringAr;            { рядки для друку таблиць }
    FileInput,                { файл вхідних документів }
    FileAdd,                  { файл документів, що додаються, }
    FileKodif,                { файл кодифікатора виробів }
    FileRes : text;           { файл результатів контролю }
    FileOut                   { архівний файл виробів }
         : file of ProductType;

Implementation

End.



UNIT BasUnit;
{ Сервісні процедури і функції }
Interface

Uses Crt,DesUnit,Printer;

Procedure WaitEnter;
```

```
Procedure PrintString(X,Y:byte; S:string);
Procedure WritelnString(S:string);
Procedure PrintKeyAndWaitEnter;
Procedure CheckPageScreen(Var j:byte);
Function  Space(S:string; k:byte):byte;
Function  NotSpace(S:string; k:byte):byte;
Function  FillString(S:string; len,pk:byte):string;
Function  GetNumber(MinNumber,MaxNumber:real;
                    m1,n1,m2,n2:byte):real;
Procedure UsesDevice;
Procedure PrintHat(n:byte);
Procedure DisposeProduct;
Procedure SortKodif(nk:byte);
Function  SearchKodif(Kod:longint; nk:byte):byte;


Implementation

{-----------------------------------------------------------}

Procedure WaitEnter;
{ Затримка виконання програми доти, }
{ поки не буде натиснута клавіша Enter }
Var   ch : char;
Begin
  Repeat
    ch:=ReadKey;
  Until ord(ch) = Enter;
End { WaitEnter };
{-----------------------------------------------------------}

Procedure PrintString(X,Y:byte; S:string);
{ Друк рядка S з позиції X рядка екрана з номером Y }
Begin
  GotoXY(X,Y); Write(S);
End { PrintString };
{-----------------------------------------------------------}

Procedure WritelnString(S:string);
{ Вивід рядка S на екран, принтер і магнітний диск }
Begin
  Writeln(S);
  Case Device of
    1 : Writeln(FileRes,S);
    2 : Writeln(Lst,S)
  end;
End { WritelnString };
{-----------------------------------------------------------}

Procedure PrintKeyAndWaitEnter;
{ Друк рядка-константи PressKey з позиції 1 рядка екрана 25  }
```

```
{ і затримка виконання програми до натискання клавіші Enter  }
Begin
  PrintString(1,25,PressKey);
  WaitEnter;
  ClrScr;
End { PrintKeyAndWaitEnter };
{-------------------------------------------------------------}

Procedure CheckPageScreen(Var j:byte);
{ Контроль розміру сторінки на екрані }
Const  LengthPage = 23;  { кіл-у рядків на одній сторінці }
       Sr = 'Наступна сторінка - "Enter" ';
Begin
  Inc(j);
  If j=LengthPage then
    Begin
      j:=0;
      PrintString(1,25,Sr);
      WaitEnter;
      ClrScr;
    End;
End { CheckPageScreen };
{-------------------------------------------------------------}

Function Space(S:string; k:byte):byte;
{ Пошук найближчого пропуску }
Var  i : byte;
Begin
  Space:=0;
  For i:=k to length(S) do
    If S[i]=' ' then
      Begin
        Space:=i; Exit
      End;
End { Space };
{-------------------------------------------------------------}

Function NotSpace(S:string; k:byte):byte;
{ Пошук найближчого непропуску }
Var  i : byte;
Begin
  NotSpace:=0;
  For i:=k to length(S) do
    If S[i]<>' ' then
      Begin
        NotSpace:=i; Exit
      End;
End { NotSpace };
{-------------------------------------------------------------}

Function FillString(S:string; len,pk:byte):string;
```

```
{ Доповнення рядка S до довжини len пропусками ліворуч }
{ (pk=0)  або праворуч (pk=1)                              }
Var  i : byte;
Begin
  If length(S)<len then
    For i:=1 to len-length(S) do
      If pk=0 then
        S:=' '+S
      Else
        S:=S+' ';
  FillString:=S;
End { FillString };
{-----------------------------------------------------------}

Function GetNumber(MinNumber,MaxNumber:real;
                   m1,n1,m2,n2:byte):real;
{Уведення числа із клавіатури; MinNumber,MaxNumber - припустимий}
{ діапазон; m1,n1,m2,n2 - формат повідомлення про діапазон      }
Var  k : integer;
    Number : real;
    Cond : boolean;
Begin
  Repeat
    Cond:=true;
    {$I-} Readln(Number); {$I+}
    k:=IOResult;
    If k<>0 then
      Begin
        Writeln(#7'Неправильний формат числа');
        Writeln('Повторіть уведення');
        Cond:=false
      End
    Else
      If (Number<MinNumber) or (Number>MaxNumber) then
        Begin
          Write(#7'Число повинне бути в діапазоні ');
          Writeln('від ',MinNumber:m1:n1,' до ',
                  MaxNumber:m2:n2,' .');
          Writeln('Повторіть уведення');
          Cond:=false
        End;
  Until Cond;
  GetNumber:=Number;
End { GetNumber };
{-----------------------------------------------------------}

Procedure UsesDevice;
{ Запит про пристрій для виводу результатів }
Begin
  Writeln('Укажіть пристрій для виводу результатів:');
  Writeln('   0 - тільки екран');
```

```
  Writeln('   1 - екран і магнітний диск');
  Writeln('   2 - екран і принтер');
  Device:=Round(GetNumber(0,2,1,0,1,0));
  ClrScr;
End { UsesDevice };
{-------------------------------------------------------}

Procedure PrintHat(n:byte);
{ Вивід шапки таблиці з n рядків }
Var  i : byte;
Begin
  For i:=1 to n do
    WritelnString(St[i]);
End { PrintHat };
{-------------------------------------------------------}

Procedure DisposeProduct;
{ Вилучення черги виробів }
Begin
  While Lp<>nil do
    Begin
      Run:=Lp;
      Dispose(Run);
      Lp:=Lp^.Next;
    End;
End { DisposeProduct };
{-------------------------------------------------------}

Procedure SortKodif(nk:byte);
{ Угруповання масиву Kodifs по зростанню компонента Kod }
Var  i,m : byte;
     Kod : longint;
    Cond : boolean;
Begin
  Cond:=true; m:=nk-1;
  While Cond do
    Begin
      Cond:=false;
      For i:=1 to m do
        Begin
          Kod:=Kodifs[i].Kod;
          If Kod>Kodifs[i+1].Kod then
            Begin
              Kodif:=Kodifs[i];
              Kodifs[i]:=Kodifs[i+1];
              Kodifs[i+1]:=Kodif;
              Cond:=true;
            End;
        End;
      Dec(m);
    End;
```

```
End { SortKodif };
{-------------------------------------------------------------}

Function SearchKodif(Kod:longint; nk:byte):byte;
{ Двійковий пошук у масиві кодифікатора KodifProducts }
{ елемента з кодом Kod                                 }
Var  i1,i2,m : byte;
Begin
  SearchKodif:=0;
  i1:=1; i2:=nk;
  While i1<=i2 do
    Begin
      m:=(i1+i2) div 2;
      If Kod=Kodifs[m].Kod then
        Begin
          SearchKodif:=m; Exit
        End
      Else
        If Kod>Kodifs[m].Kod then
          i1:=m+1
        Else
          i2:=m-1;
    End;
End { SearchKodif };
{-------------------------------------------------------------}


End.

UNIT InpUnit;
{ Контроль коректності вхідних текстових файлів }

Interface

Uses Crt,DesUnit,BasUnit,Printer;

Procedure CheckFiles;

Implementation

{-------------------------------------------------------------}
Const
   NfMax = 200;    { макс. кількість рядків у текстовому файлі }
Type
   FileStringAr = array[1..NfMax] of string80;
   FileStringArPtr = ^FileStringAr;
   ProductAr = array[1..NfMax] of ProductType;
   ProductArPtr = ^ProductAr;
Var
   np,              { кількість рядків у файлі FileProductInput }
   nd,              { кількість рядків у файлі FileAddProduct }
```

```
   nk : byte;       { кількість рядків у файлі FileKodifProduct }
   FatalError            { наявність помилки у вхідних даних }
      : boolean;
   Sr : string80;        { повідомлення про помилку }
   Sf : FileStringArPtr; { масив рядків текстового файлу }
   Products,
   AddProducts : ProductArPtr;
   FileError             { файл протоколу перевірок }
      : file of string80;
{-----------------------------------------------------------}

Procedure ReadAndCheckSpaces(Var F:text; FileName:string80;
                             Var nf:byte; Nmax:byte);
{ Уведення текстового файлу і вилучення в ньому порожніх рядків }
Var  i,j,k : byte;
   kf : integer;
   SignSpace : boolean;      { ознака порожнього рядка у файлі }
   S : string80;
Begin

{ Перевірка наявності файлу F з ім'ям FileName }
  {$I-} Reset(F); {$I+}
  k:=IOResult;
  If k<>0 then
    Begin
      FatalError:=true;
      Sr:='Відсутній вихідний файл '+FileName;
      Write(FileError,Sr);
      Exit;
    End;

{ Читання текстового файлу }
  nf:=0;
  While not eof(F) do
    Begin
      Inc(nf);
      Readln(F,Sf^[nf]);
    End;
  Close(F);

{ Перевірка: чи є файл F порожнім }
  If nf=0 then
    Begin
      FatalError:=true;
      Sr:='Вихідний файл '+FileName+' порожній';
      Write(FileError,Sr);
      Exit;
    End;

{ Перевірка: чи перевищує кількість рядків макс.значення }
  If nf>Nmax then
```

```
    Begin
      FatalError:=true;
      Str(Nmax,S);
      Sr:='У вихідному файлі '+FileName+' понад '+S+' рядки';
      Write(FileError,Sr);
      Exit;
    End;

{ Вилучення порожніх рядків із складу файлу F }
  SignSpace:=false;
  i:=1;
  While i<=nf do
    Begin
      k:=NotSpace(Sf^[i],1);
      If k=0 then
        Begin
          For j:=i to nf-1 do
            Sf^[j]:=Sf^[j+1];
          Dec(nf);
          SignSpace:=true;
        End
      Else
        Inc(i);
    End;
  If SignSpace then
    Begin
      Rewrite(F);
      For i:=1 to nf do
        Writeln(F,Sf^[i]);
      Close(F);
    End;
End { ReadAndCheckSpaces };
{-------------------------------------------------------------}

Procedure FormatFileProduct(Var F:text; FileName:string80;
               Var nf:byte; Nmax:byte; Var Prod:ProductArPtr);
{ Перевірка форматів файлів FileInput і FileAdd }
Var  k,k1,k2 : byte;
           i : word;
        Code : integer;
        Cond : boolean;
       Sa,Sb,S1,S2 : string80;
Begin
  ReadAndCheckSpaces(F,FileName,nf,Nmax);
  If FatalError then
    Exit;
  For i:=1 to nf do
    Begin
      Sa:=Sf^[i];
      With Product do
        Begin
```

```
Cond:=true; k2:=0; k:=0;
While Cond do
  Begin
    k1:=NotSpace(Sa,k2+1);
    If k1=0 then
      Cond:=false
    Else
      Begin
        k2:=Space(Sa,k1+1);
        If k2=0 then
          Begin
            k2:=length(Sa)+1; Cond:=false
          End;
        Inc(k);
        If k>8 then
          Begin
            FatalError:=true;
            Str(i,S1);
            Sr:='Файл '+FileName+': у рядку '+S1+
                'понад 8 елементів';
            Write(FileError,Sr);
          End;
        Sb:=Copy(Sa,k1,k2-k1);
        Case k of
          1 : Val(Sb,NumberShop,Code);
          2 : Val(Sb,Kod,Code);
          3 : Begin
                If length(Sb)>5 then
                  Begin
                    FatalError:=true;
                    Str(i,S1);
                    Sr:='Файл '+FileName+': у рядку '
                      +S1+' довжина елемента 3 понад '+
                      '5 символів';
                    Write(FileError,Sr);
                  End;
                Dimens:=FillString(Sb,5,1);
              End;
          4 : Val(Sb,Price,Code);
          5 : Val(Sb,Plan[1],Code);
          6 : Val(Sb,Plan[2],Code);
          7 : Val(Sb,Fact[1],Code);
          8 : Val(Sb,Fact[2],Code);
        end;
        If (k<>3) and (Code<>0) then
          Begin
            FatalError:=true;
            Str(i,S1); Str(k,S2);
            Sr:='Файл '+FileName+': у рядку '+S1+
                ' неправильний формат елемента '+S2+
                ' ('+Sb+')';
```

```
                    Write(FileError,Sr);
                  End;
              End;
          End;
        If k<8 then
          Begin
            FatalError:=true;
            Str(i,S1);
            Sr:='Файл '+FileName+': у рядку '+S1+
                'менше 8 елементів';
            Write(FileError,Sr);
          End;
      End;
    Prod^[i]:=Product;
  End;
End { FormatFileProduct };
{-------------------------------------------------------------}

Procedure FormatFileKodif;
{ Перевірка форматів файлу FileKodif }
Var  k,k1,k2 : byte;
          i : word;
       Code : integer;
       Cond : boolean;
     Sa,Sb,S1,S2 : string80;
Begin
  ReadAndCheckSpaces(FileKodif,'Kodif.txt',nk,MaxKodif);
  If FatalError then
    Exit;
  For i:=1 to nk do
    Begin
      Sa:=Sf^[i];
      With Kodif do
        Begin
          Cond:=true; k2:=0; k:=0;
          While Cond do
            Begin
              k1:=NotSpace(Sa,k2+1);
              If k1=0 then
                Cond:=false
              Else
                Begin
                  k2:=Space(Sa,k1+1);
                  If k2=0 then
                    Begin
                      k2:=length(Sa)+1; Cond:=false
                    End;
                  Inc(k);
                  If k>3 then
                    Begin
                      FatalError:=true;
```

```
            Str(i,S1);
            Sr:='Файл Kodif.txt: у рядку '+S1+
                'понад 2 елементів';
            Write(FileError,Sr);
          End;
        Sb:=Copy(Sa,k1,k2-k1);
        Case k of
          1 : Begin
                Val(Sb,Kod,Code);
                If Code<>0 then
                  Begin
                    FatalError:=true;
                    Str(i,S1);
                    Sr:='Файл Kodif.txt: у рядку '+S1+
                      ' неправильний формат елемента 1'+
                        ' ('+Sb+')';
                    Write(FileError,Sr);
                  End;
              End;
          2 : Begin
                If length(Sb)>35 then
                  Begin
                    FatalError:=true;
                    Str(i,S1);
                    Sr:='Файл Kodif.txt: у рядку '+S1+
                        ' довжина елемента 2 понад 35 '+
                        ' символів';
                    Write(FileError,Sr);
                  End;
                Name:=FillString(Sb,35,1);
              End;
          3 : Begin
                While Name[length(Name)]=' ' do
                  Delete(Name,length(Name),1);
                Sb:=Name+' '+Sb;
                If length(Sb)>35 then
                  Begin
                    FatalError:=true;
                    Str(i,S1);
                    Sr:='Файл Kodif.txt: у рядку '+S1+
                        ' довжина елемента 2 понад 35 '+
                        ' символів';
                    Write(FileError,Sr);
                  End;
                Name:=FillString(Sb,35,1);
              End;
        end;
      End;
  End;
If k<2 then
  Begin
```

```
              FatalError:=true;
              Str(i,S1);
              Sr:='Файл Kodif.txt: у рядку '+S1+
                  'менше 2 елементів';
              Write(FileError,Sr);
            End;
        End;
      Kodifs[i]:=Kodif;
    End;
End { FormatFileKodif };
{-------------------------------------------------------------}

Procedure CheckProdDiapason(FileName:string80; nf:byte;
                            Prod:ProductArPtr);
{ Перевірка діапазонів параметрів в FileInput і FileAdd }
Var  i : byte;
   S1,S2,S3,S4 : string80;
{ ----------------------------- }
Procedure ReportError1(i,k:word; d1,d2:longint);
Begin
  FatalError:=true;
  Str(i,S1); Str(k,S2); Str(d1,S3); Str(d2,S4);
  Sr:='Файл '+FileName+': у рядку '+S1+' елемент '+S2+
      ' поза межами '+S3+'..'+S4;
  Write(FileError,Sr);
End { ReportError1};
{ ----------------------------- }
Procedure ReportError2(i,k:word; d1,d2:real);
Begin
  FatalError:=true;
  Str(i,S1); Str(k,S2); Str(d1:8:1,S3); Str(d2:8:1,S4);
  While S3[1]=' ' do
    Delete(S3,1,1);
  While S4[1]=' ' do
    Delete(S4,1,1);
  Sr:='Файл '+FileName+': у рядку '+S1+' елемент '+S2+
      ' поза межами '+S3+'..'+S4;
  Write(FileError,Sr);
End { ReportError2};
{ ----------------------------- }
Begin
  For i:=1 to nf do
    With Prod^[i] do
      Begin
        If (NumberShop<1) or (NumberShop>99) then
          ReportError1(i,1,1,99);
        If (Kod<100000) or (Kod>999999) then
          ReportError1(i,2,100000,999999);
        If (Price<0.1) or (Price>1000) then
          ReportError2(i,4,0.1,100);
        If (Plan[1]<10) or (Plan[1]>10000) then
```

```
          ReportError1(i,5,10,10000);
        If (Plan[2]<10) or (Plan[2]>10000) then
          ReportError1(i,6,10,10000);
        If (Fact[1]<10) or (Fact[1]>10000) then
          ReportError1(i,7,10,10000);
        If (Fact[2]<10) or (Fact[2]>10000) then
          ReportError1(i,8,10,10000);
      End;
End { CheckProdDiapason };
{-------------------------------------------------------------}

Procedure CheckKodifDiapason;
Var  i : byte;
    S1 : string80;
Begin
  For i:=1 to nk do
    With Kodifs[i] do
      If (Kod<100000) or (Kod>999999) then
        Begin
          FatalError:=true;
          Str(i,S1);
          Sr:='Файл Kodif.txt: у рядку '+S1+' елемент 1'+
              ' поза межами 100000..999999';
          Write(FileError,Sr);
        End;
End { CheckKodifDiapason };
{-------------------------------------------------------------}

Procedure ProdParameters(Prod:ProductArPtr; n:byte;
                         FileName:string80);
{ Перевірка параметрів записів у файлах FileInput і FileAdd }
Const  Measurs : array[1..2] of string[5] = ('шт.  ','кг   ');
Var  i,j,k : byte;
     Kod : longint;
    Cond : boolean;
    Meas : string[5];
    S1,S2,S3 : string80;
Begin

{ Перевірка дублювання параметра KodProduct }
  For i:=1 to n-1 do
    Begin
      Kod:=Prod^[i].Kod;
      For j:=i+1 to n do
        If Kod=Prod^[j].Kod then
          Begin
            FatalError:=true;
            Str(i,S1); Str(j,S2); Str(Kod,S3);
            Sr:='Файл '+FileName+': рівні значення KodProduct '+
                'у рядках '+S1+' і '+S2+' ('+S3+')';
            Write(FileError,Sr);
```

```
          End;
    End;

{ Перевірка наявності параметра KodProduct у кодифікаторі }
  For i:=1 to n do
    Begin
      Kod:=Prod^[i].Kod;
      k:=SearchKodif(Kod,nk);
      If k=0 then
        Begin
          FatalError:=true;
          Str(i,S1); Str(Kod,S2);
          Sr:='Файл '+FileName+': код виробу '+S2+' (рядок '+S1+
              ') відсутній у кодифікаторі';
          Write(FileError,Sr);
        End;
    End;

{ Перевірка одиниць розмірності }
  For i:=1 to n do
    Begin
      Meas:=Prod^[i].Dimens;
      Cond:=false;
      For j:=1 to 2 do
        If Meas=Measurs[j] then
          Cond:=true;
      If not Cond then
        Begin
          FatalError:=true;
          Str(i,S1);
          While Meas[length(Meas)]=' ' do
            Delete(Meas,length(Meas),1);
          Sr:='Файл '+FileName+': у рядку '+S1+' неправильна '+
              'од.розм. ('+Meas+')';
          Write(FileError,Sr);
        End;
    End;

End { ProdParameters };
{-------------------------------------------------------------}

Procedure KodifParameters;
{ Перевірка дублювання параметра Kod у файлі FileKodif }
Var  i,j,k : byte;
       Kod : longint;
     S1,S2,S3 : string80;
Begin
  For i:=1 to nk-1 do
    Begin
      Kod:=Kodifs[i].Kod;
      For j:=i+1 to nk do
```

```
        If Kod=Kodifs[j].Kod then
          Begin
            FatalError:=true;
            Str(i,S1); Str(j,S2); Str(Kod,S3);
            Sr:='Файл Kodif.txt: рівні значення KodProduct '+
                'у рядках '+S1+' і '+S2+' ('+S3+')';
            Write(FileError,Sr);
          End;
    End;
End { KodifParameters };
{-------------------------------------------------------------}

Procedure ReadFileError;
{ Читання протоколу контролю текстових файлів }
Begin
  ClrScr;
  Seek(FileError,0);
  While not eof(FileError) do
    Begin
      Read(FileError,Sr);
      Writeln(Sr);
    End;
  PrintKeyAndWaitEnter;
End { ReadFileError };
{-------------------------------------------------------------}

Procedure CheckFiles;
{ Контроль коректності текстових файлів }
Var  i : byte;
Begin
  Assign(FileError,'Error.dat');
  Rewrite(FileError);
  Sr:=' ';
  Write(FileError,Sr);
  Sr:='         ПРОТОКОЛ КОНТРОЛЮ ТЕКСТОВИХ ФАЙЛІВ';
  Write(FileError,Sr);
  Sr:=' ';
  Write(FileError,Sr);
  FatalError:=false;
  New(Sf);
  New(Products); New(AddProducts);
  FormatFileProduct(FileInput,'Input.txt',np,Nfmax,
                    Products);
  FormatFileProduct(FileAdd,'Add.txt',nd,Nfmax,
                    AddProducts);
  FormatFileKodif;
  If not FatalError then
    Begin
      CheckProdDiapason('Input.txt',np,Products);
      CheckProdDiapason('Add.txt',nd,AddProducts);
      CheckKodifDiapason;
```

```
    End;

  If not FatalError then
    Begin
      SortKodif(nk);
      ProdParameters(Products,np,'Input.txt');
      ProdParameters(AddProducts,nd,'Add.txt');
      KodifParameters;
    End;

  Sr:=' ';
  Write(FileError,Sr);
  If FatalError then
    Sr:='Скорегуйте вхідні файли'
  Else
    Sr:='У вхідних файлах помилок не виявлеое';
  Write(FileError,Sr);
  ReadFileError;
  Dispose(Sf); Sf:=nil;
  Dispose(Products);    Products:=nil;
  Dispose(AddProducts); AddProducts:=nil;
  Close(FileError);
  Erase(FileError);
End { CheckFiles };
{-------------------------------------------------------------}

End.

UNIT WorkUnit;
{ Обробні процедури і функції }

Interface

Uses Crt,DesUnit,BasUnit,Printer;

Procedure CreateArchive;
Procedure SortArchive;
Procedure PrintArchive;
Procedure PrintKodif;
Procedure AddArchive;
Procedure DeleteArchive;
Procedure ChangeArchive;
Procedure WorkUpArchive;


Implementation

{-------------------------------------------------------------}

Procedure ReadProduct(Var F:text);
{ Уведення з текст.файлу FileInput або FileAdd компонента архіву }
```

```
Var  k,k1,k2 : byte;
        Code : integer;
        Cond : boolean;
       Sa,Sb : string80;
Begin
  With Product do
    Begin
      Readln(F,Sa);
      Cond:=true; k2:=0; k:=0;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1; Cond:=false
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Val(Sb,NumberShop,Code);
                2 : Val(Sb,Kod,Code);
                3 : Dimens:=FillString(Sb,5,1);
                4 : Val(Sb,Price,Code);
                5 : Val(Sb,Plan[1],Code);
                6 : Val(Sb,Plan[2],Code);
                7 : Val(Sb,Fact[1],Code);
                8 : Val(Sb,Fact[2],Code);
              end;
            End;
        End;
    End;
End { ReadProduct };
{-------------------------------------------------------------}

Procedure ReadKodif;
{ Уведення з текстового файлу FileKodif компонента кодифікатора }
Var  k,k1,k2 : byte;
        Code : integer;
        Cond : boolean;
       Sa,Sb : string80;
Begin
  With Kodif do
    Begin
      Readln(FileKodif,Sa);
      Cond:=true; k2:=0; k:=0;
      While Cond do
```

```
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1; Cond:=false
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Val(Sb,Kod,Code);
                2 : Name:=FillString(Sb,35,1);
                3 : Begin
                      While Name[length(Name)]=' ' do
                        Delete(Name,length(Name),1);
                      Name:=Name+' '+Sb;
                      Name:=FillString(Name,35,1);
                    End;
              end;
            End;
        End;
    End;
End { ReadKodif };
{-------------------------------------------------------}

Procedure MakeKodifs;
{ Читання файлу FileKodif і формування масиву Kodifs }
Begin
  Reset(FileKodif);
  nk:=0;
  While not eof(FileKodif) do
    Begin
      Inc(nk); ReadKodif;
      Kodifs[nk]:=Kodif;
    End;
  Close(FileKodif);
End { MakeKodifs };
{-------------------------------------------------------}

Procedure CreateArchive;
{ Створення архівного файлу виробів і масиву Kodifs }
Begin
  Reset(FileInput); Rewrite(FileOut);
  While not eof(FileInput) do
    Begin
      ReadProduct(FileInput);
      Write(FileOut,Product);
```

```
    End;
  Close(FileInput); Close(FileOut);
  SignArchive:=true;
  MakeKodifs; SortKodif(nk);
  Writeln('Создание архіву закінчене');
  PrintKeyAndWaitEnter;
End { CreateArchive };
{-------------------------------------------------------}

Procedure ReadFileOut;
{ Читання з архіву і формування черги записів }
Begin
  Reset(FileOut);
  np:=0;
  Lp:=nil; Rp:=nil;
  While not eof(FileOut) do
    Begin
      Read(FileOut,Product);
      New(Run); Inc(np);
      Run^.Inf:=Product;
      Run^.Next:=nil;
      If Lp=nil then
        Lp:=Run
      Else
        Rp^.Next:=Run;
      Rp:=Run;
    End;
  Close(FileOut);
End { ReadFileOut };
{-------------------------------------------------------}

Procedure WriteFileOut;
{ Пересилання черги в архів з наступним її вилученням }
Var  i : word;
Begin
  Rewrite(FileOut);
  Run:=Lp;
  While Run<>nil do
    Begin
      Write(FileOut,Run^.Inf);
      Run:=Run^.Next
    End;
  Close(FileOut);
  DisposeProduct;
End { WriteFileOut };
{-------------------------------------------------------}

Procedure SortArchive;
{ Сортування архіву по зростанню коду виробу }
Var    i,m : word;
       KodMax : longint;
```

```
       Pmax : PointerProduct;
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  m:=np;
  While m>1 do
    Begin
      Run:=Lp;
      KodMax:=Run^.Inf.Kod; Pmax:=Run;
      For i:=2 to m do
        Begin
          Run:=Run^.Next;
          If Run^.Inf.Kod>KodMax then
            Begin
              KodMax:=Run^.Inf.Kod; Pmax:=Run
            End;
        End;
      If Pmax<>Run then
        Begin
          Product:=Run^.Inf; Run^.Inf:=Pmax^.Inf;
          Pmax^.Inf:=Product;
        End;
      Dec(m);
    End;
  WriteFileOut;
  Writeln('Сортировка архіву закінчена');
  PrintKeyAndWaitEnter;
End { SortArchive };
{-------------------------------------------------------------}

Procedure PrintArchive;
{ Вивід архіву виробів на екран, принтер і магнітний диск }
Var   i : word;
      j : byte;
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  Reset(FileOut);
  St[1]:='     АРХІВ ВІДОМОСТЕЙ ПРО ПРОДУКЦІЮ, ЩО ВИПУСКАЄТЬСЯ ';
  St[2]:='-------------------------------------------------------';
St[3]:': N  : N  :  Код  : Ед. :       :  План випуску :'+
         '  Факт.випуск  :';
```

```
  St[4]:=':п/п :цеха:изделия: изм.: Ціна  +-i-i-i-T--i-i-i+'+
         '-i-i-i-T--i-i-i+';
  St[5]:=':    :    :       :     :       :   1:2   :'+
         '   1:2   :';
  St[6]:='+-i-i+-i-i+-i-i-i-+-i-i-+-i-i-i-+-i-i-i-+-i-i-i-+'+
         '-i-i-i-+-i-i-i-+';
  PrintHat(6);
  j:=6; i:=0;
  While not eof(FileOut) do
    Begin
      Read(FileOut,Product);
      Inc(i); Str(i:3,St[1]);
      With Product do
        Begin
          Str(NumberShop:3,St[2]); Str(Kod:6,St[3]);
          St[4]:=Dimens;  Str(Price:6:2,St[5]);
          Str(Plan[1]:6:0,St[7]); Str(Plan[2]:6:0,St[8]);
          Str(Fact[1]:6:0,St[9]); Str(Fact[2]:6:0,St[10]);
        End;
      St[1]:=':'+St[1]+'. '+St[2]+'   '+St[3]+'   '+St[4]+St[5]+
             ' '+St[7]+' '+St[8]+'  '+St[9]+'  '+St[10]+'  :';
      WritelnString(St[1]);
      CheckPageScreen(j);
    End;
  St[6]:='L--i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-'+
         '-i-i-i-i-i-i-i';
  WritelnString(St[6]);
  Close(FileOut);
  PrintKeyAndWaitEnter;
End { PrintArchive };
{-------------------------------------------------------------}

Procedure PrintKodif;
{ Вивід кодифікатора виробів на екран, принтер і магнітний диск }
Var   i,j : byte;
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  St[1]:='         КОДИФІКАТОР ПРОДУКЦІЇ, ЩО ВИПУСКАЄТЬСЯ';
  St[2]:='-i-i-i-T---------------T-i-i-i-i-i-i-i-i-i-i-i-i-i-i'+
         '┗━━━━━━━━━━━━━━━┓'
St[3]:=': Nп/п :   Код виробу  :   Найменування  виробу  '+
         '              :';
  St[4]:='+-i-i-i+-i-i-i-i-i-i-i-i+-i-i-i-i-i-i-i-i-i-i-i-i-i-i'+
         '-i-i-i-i-i-i-i+';
  PrintHat(4);
  j:=4;
```

```
  For i:=1 to nk do
    Begin
      Kodif:=Kodifs[i];
      Str(i:4,St[1]); Str(Kod.Kod:6,St[2]);
      St[1]:=':'+St[1]+'.       '+St[2]+'         '+
             Kodif.Name+'  :';
      WritelnString(St[1]);
      CheckPageScreen(j);
    End;
  St[4]:='L--i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-'+
         '-i-i-i-i-i-i-i-';
  WritelnString(St[4]);
  PrintKeyAndWaitEnter;
End { PrintKodif };
{-------------------------------------------------------}

Procedure AddArchive;
{ Додавання компонента в архів виробів }
Label  10;
Var  Cond : boolean;
     Sr : string;
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  Reset(FileAdd);
  While not eof(FileAdd) do
    Begin
      ReadProduct(FileAdd);
      Str(Product.Kod,Sr);
      Run:=Lp; Cond:=false;
      While Run<>nil do
        Begin
          If Run^.Inf.Kod=Product.Kod then
            Begin
              Cond:=true; Goto 10
            End;
          Run:=Run^.Next;
        End;
      10:
      If not Cond then
        Begin
          New(Run); Inc(np);
          Run^.Inf:=Product;
          Run^.Next:=nil;
          If Lp=nil then
            Lp:=Run
```

```
          Else
            Rp^.Next:=Run;
          Rp:=Run;
        End
      Else
        WritelnString('В архіві вже є виріб з кодом '+Sr);
    End;
  WriteFileOut;
  Close(FileAdd);
  Writeln('Доповнення архіву закінчене');
  PrintKeyAndWaitEnter;
End { AddArchive };
{-------------------------------------------------------}

Procedure DeleteArchive;
{ Вилучення компонента з архіву виробів }
Label  10;
Var    Kod : longint;
      Cond : boolean;
       Del : PointerProduct;
        Sr : string;
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  ClrScr;
  WritelnString('Укажіть код компонента, що вилучається,');
  Kod:=Round(GetNumber(0,999999,1,0,6,0));
  Str(Kod,Sr); WritelnString('Kod = '+Sr);
  Cond:=false;
  If Kod=Lp^.Inf.Kod then
    Begin
      Cond:=true;
      Run:=Lp; Lp:=Lp^.Next;
      Dispose(Run);
    End
  Else
    Begin
      Run:=Lp;
      While Run^.Next<>nil do
        Begin
          If Kod=Run^.Next^.Inf.Kod then
            Begin
              Cond:=true;
              Del:=Run^.Next;
              If Run^.Next=Rp then
                Rp:=Run;
```

```
              Run^.Next:=Del^.Next;
              Dispose(Del);
              Goto 10
            End;
          Run:=Run^.Next
        End;
    End;
  10:
  If Cond then
    Begin
      Dec(np);
      WriteFileOut;
      Writeln('Вилучення компонента з архіву закінчене');
    End
  Else
    WritelnString('В архіві немає компонента з кодом '+Sr);
  PrintKeyAndWaitEnter;
End { DeleteArchive };
{-------------------------------------------------------------}

Procedure ChangeArchive;
{ Зміна компонента в архіві виробів }
Label  10;
Var  Kod : longint;
    Cond : boolean;
      Sr : string;
{ ------------------------------- }
Procedure MakeComponent;
{ Формування змінюваного компонента }
Var  k,k1,k2 : byte;
       Code : integer;
       Cond : boolean;
      Sa,Sb : string80;
Begin
  With Product do
    Begin
      Readln(Sa);
      WritelnString(Sa);
      Cond:=true; k2:=0; k:=0;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1; Cond:=false
                End;
```

```
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Dimens:=FillString(Sb,5,1);
                2 : Val(Sb,Price,Code);
                3 : Val(Sb,Plan[1],Code);
                4 : Val(Sb,Plan[2],Code);
                5 : Val(Sb,Fact[1],Code);
                6 : Val(Sb,Fact[2],Code);
              end;
            End;
        End;
    End;
End { MakeComponent };
{ ---------------------------------------- }
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  ClrScr;
  WritelnString('Укажіть код виробу змінюваного компонента');
  Kod:=Round(GetNumber(0,999999,1,0,6,0));
  Str(Kod,Sr); WritelnString('Kod = '+Sr);
  Run:=Lp; Cond:=false;
  While Run<>nil do
    Begin
      If Kod=Run^.Inf.Kod then
        Begin
          Cond:=true;
          Product:=Run^.Inf;
          WritelnString('       Укажіть наступні реквізити :');
          WritelnString('од.виміру  ціна  план-1  план-2 '+
                        ' факт-1  факт-2');
          MakeComponent;
          Run^.Inf:=Product;
          WriteFileOut;
          Writeln('Зміна компонента в архіві закінчене');
          Goto 10
        End;
      Run:=Run^.Next;
    End;
  10:
  If not Cond then
    WritelnString('В архіві немає компонента з кодом '+Sr);
  PrintKeyAndWaitEnter;
End { ChangeArchive };
{-------------------------------------------------------------}
```

```
Procedure WorkUpArchive;
{ Обробка архіву виробів }
Var      i : word;            { параметр циклу }
     Shop,                    { номер цеху }
     j,k,m : byte;
       Kod : longint;         { код виробу }
    PlanSt1,PlanSt2,PlanGod, { сумарні планові показники }
    FactSt1,FactSt2,FactGod, { сумарні фактичні показники }
    ProcSt1,ProcSt2,ProcGod, { відсоток виконання плану }
    BufSt         : real;    { буферна змінна }
Begin
  If not SignArchive then
    Begin
      Writeln('Архівний файл не створений. Режим відміняється');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  Writeln('Укажіть номер цеху');
  Shop:=Round(GetNumber(1,99,1,0,2,0));
  PlanSt1:=0; PlanSt2:=0;
  FactSt1:=0; FactSt2:=0;
  k:=0; Run:=Lp;
  While Run<>nil do
    Begin
      If Shop=Run^.Inf.NumberShop then
        Begin
          Inc(k);
          BufSt:=Run^.Inf.Plan[1]*Run^.Inf.Price;
          PlanSt1:=PlanSt1+BufSt;
          BufSt:=Run^.Inf.Plan[2]*Run^.Inf.Price;
          PlanSt2:=PlanSt2+BufSt;
          BufSt:=Run^.Inf.Fact[1]*Run^.Inf.Price;
          FactSt1:=FactSt1+BufSt;
          BufSt:=Run^.Inf.Fact[2]*Run^.Inf.Price;
          FactSt2:=FactSt2+BufSt;
        End;
      Run:=Run^.Next;
    End;
  If k=0 then
    Writeln('В архіві немає відомостей про цех ',Shop)
  Else
    Begin
      PlanGod:=PlanSt1+PlanSt2;
      FactGod:=FactSt1+FactSt2;
      ProcSt1:=100*FactSt1/PlanSt1;
      ProcSt2:=100*FactSt2/PlanSt2;
      ProcGod:=100*FactGod/PlanGod;
      ClrScr; Str(Shop:2,St[1]);
      St[1]:='             ВІДОМОСТІ ПРО ВИКОНАННЯ ПЛАНУ ПО ЦЕХУ '
```

```
              +St[1];
       St[2]:='-i-i-i-i-i-i-i-i-i-i-T--i-i-i-i-i-i-i-i-i-i'+
              '└─────────┴──────────────┘'
St[3]:=':                    :          Півріччя    '+
              '       :        За      :';
       St[4]:=':                     +-i-i-i-i-i-i-iT--i-'+
              '-i-i-i-i-i+     рік     :';
       St[5]:=':                     :       1      :    '+
              '  2       :             :';
       St[6]:='+-i-i-i-i-i-i-i-i-i-i-i+-i-i-i-i'+
              '-i-i-i+-i-i-i-i-i-i-i-i+-i-i-i-i-i-i-+';
       PrintHat(6);
       Str(PlanSt1:10:2,St[1]); Str(PlanSt2:10:2,St[2]);
       Str(PlanGod:10:2,St[3]);
       St[1]:=': План випуску       :  '+St[1]+'  :  '+St[2]+
              '  :  '+St[3]+' :';
       WritelnString(St[1]);
       Str(FactSt1:10:2,St[1]); Str(FactSt2:10:2,St[2]);
       Str(FactGod:10:2,St[3]);
       St[1]:=': Фактичний випуск :  '+St[1]+'  :  '+St[2]+
              '  :  '+St[3]+' :';
       WritelnString(St[1]);
       Str(ProcSt1:8:2,St[1]); Str(ProcSt2:8:2,St[2]);
       Str(ProcGod:8:2,St[3]);
       St[1]:=': Відсоток виконання :    '+St[1]+'  :    '+
              St[2]+'  :    '+St[3]+' :';
       WritelnString(St[1]);
       St[6]:='L--i-i-i-i-i-i-i-i-i-i-+-i-i-i-i-i-i-i-i+-i-i-i-i-i-i-
i+'+
              '-i-i-i-i-i-i-i';
       WritelnString(St[6]);
     End;
  PrintKeyAndWaitEnter;
  St[1]:=' ';
  St[2]:=' СПИСОК ВИРОБІВ, ПО ЯКИМ НЕ ВИКОНАНИЙ РІЧНИЙ '+
         'ПЛАН ВИРОБНИЦТВА';
  St[3]:='-i-i-iT----------T-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-'+
         '└─┴──────────┘'
St[4]:=':N п/п:   Код    :       Найменування виробу        '+
         ' :выполнение:';
  St[5]:=':     : виробу  :                                  '+
         ' : плану, % :';
  St[6]:='+-i-i-+-i-i-i-i-i+-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-'+
         '-+-i-i-i-i-i+';
  PrintHat(6);
  j:=6; m:=0;
  Run:=Lp;
  While Run<>nil do
    Begin
      ProcGod:=100*(Run^.Inf.Fact[1]+Run^.Inf.Fact[2])/
                   (Run^.Inf.Plan[1]+Run^.Inf.Plan[2]);
```

```
      If ProcGod<100 then
        Begin
          St[1]:=''; Inc(m);
          Kod:=Run^.Inf.Kod;
          k:=SearchKodif(Kod,nk);
          If k>0 then
            St[1]:=Kodifs[k].Name
          Else
            St[1]:='                                              ';
          Str(m:2,St[2]); Str(Kod:6,St[3]);
          Str(ProcGod:8:2,St[4]);
          St[2]:=':  '+St[2]+' :  '+St[3]+'  : '+St[1]+':'+
                 St[4]+'  :';
          WritelnString(St[2]);
          CheckPageScreen(j);
        End;
      Run:=Run^.Next;
    End;
  St[6]:='L--i-i+-i-i-i-i-i+-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-i-'+
         '-+-i-i-i-i-i-';
  WritelnString(St[6]);
  Writeln('Обробка архіву закінчена');
  PrintKeyAndWaitEnter;
End { WorkUpArchive };
{-------------------------------------------------------}

End.


Program Kontr4;
{ Створення, друк, корекція і обробка архіву відомостей }
{ про продукцію, що випускається цехами підприємства    }

Uses Crt,DesUnit,BasUnit,InpUnit,WorkUnit,Printer;

Begin

  Assign(FileInput,'E:\studies\zaoch\examples\kontr4\Input.txt');
  Assign(FileAdd,'E:\studies\zaoch\examples\kontr4\Add.txt');
  Assign(FileKodif,'E:\studies\zaoch\examples\kontr4\Kodif.txt');
  Assign(FileRes,'E:\studies\zaoch\examples\kontr4\Res.txt');
  Assign(FileOut,'E:\studies\zaoch\examples\kontr4\Out.dat');

  ClrScr;
  UsesDevice;
  If Device=1 then
    Rewrite(FileRes);
  SignArchive:=false;
  Repeat
    ClrScr;
    PrintString(15,7,'Укажіть режим роботи програми:');
```

```
    PrintString(17,8,'0 - кінець роботи;');
    PrintString(17,9,'1 - контроль текстових файлів;');
    PrintString(17,10,'2 - створення архіву виробів;');
    PrintString(17,11,'3 - сортування компонентів архіву;');
    PrintString(17,12,'4 - друк архіву виробів;');
    PrintString(17,13,'5 - друк кодифікатора виробів;');
    PrintString(17,14,'6 - додавання компонента в архів;');
    PrintString(17,15,'7 - вилучення компонента з архіву;');
    PrintString(17,16,'8 - зміна компонента в архіві;');
    PrintString(17,17,'9 - обробка архіву');
    Writeln;
    KeyRegime:=Round(GetNumber(0,9,1,0,1,0));
    Case KeyRegime of
      0 :               ;
      1 : CheckFiles    ;
      2 : CreateArchive ;
      3 : SortArchive   ;
      4 : PrintArchive  ;
      5 : PrintKodif    ;
      6 : AddArchive    ;
      7 : DeleteArchive ;
      8 : ChangeArchive ;
      9 : WorkUpArchive
      Else KeyRegime:=0;
    end;
  Until KeyRegime=0;
  If Device=1 then
    Close(FileRes);

End.
```

Результати роботи програми Kontr4
(таблиці приводяться в частковому вигляді)

1. Друк архіву після відпрацьовування режиму 2 (архів не згрупований).

АРХІВ ВІДОМОСТЕЙ ПРО ПРОДУКЦИЮ, ЩО ВИПУСКАЄТЬСЯ

N п/п	N цеху	Код виробу	Один. вим.	Ціна	План випуску 1	План випуску 2	Факт.випуск 1	Факт.випуск 2
1.	1	320011	шт.	221.50	200	180	180	260
2.	1	324531	шт.	35.14	500	600	510	610
3.	5	621412	шт.	53.14	440	460	442	434
4.	5	623462	шт.	35.30	3200	2900	3060	3070
5.	14	386410	шт.	15.60	400	600	400	410
6.	3	410410	шт.	8.20	2500	2600	2100	2200
7.	3	410411	шт.	9.60	1100	1200	1170	1190

```
:  8.  12  529270   шт.  445.20    500    700     510     710  :
:  9.  12  521570   шт.    5.80   5000   8000    6600    7500  :
: 10.  12  530510   шт.   19.80   8200   8200    8050    8130  :
: ..................................................................... :
: 25.   5  620405   шт.  890.00    160    170     162     155  :
: 26.   5  611411   шт.  876.90    110    120     123     125  :
: 27.   5  601515   шт.   41.40    560    600     578     588  :
: 28.   5  622277   шт.   28.60   1800   2000    1810    1856  :
-----------------------------------------------------------------
```

3. Друк кодифікатора виробів (режим 5).

```
                КОДИФІКАТОР ПРОДУКЦІЇ, ЩО ВИПУСКАЄТЬСЯ
-----------------------------------------------------------
: N п/п :   Код виробу  :   Найменування  виробу            :
-----------------------------------------------------------
:   1.       212233         Сідловина                      :
:   2.       320011         Кронштейн                      :
:   3.       324531         Піддон циліндра                :
:   4.       337288         Ресивер                        :
:   5.       338666         Кільце обтискне                :
:   6.       385005         Корпус двигуна                 :
:   7.       385571         Станина                        :
:   8.       385629         Корпус насоса                  :
:   9.       386388         Маховик                        :
:  10.       386410         Заготівля вала                 :
:  ..................................................       :
:  36.       623462         Склянка конусний               :
:  37.       623466         Склянка циліндричний           :
:  38.       759003         Прокладка                      :
:  39.       769344         Пластина Ф1305                 :
:  40.       780088         Заповнювач І3154а              :
:  41.       799444         Ферропласт                     :
:  42.       899444         Втулка конічна                 :
-----------------------------------------------------------
```

4. Друк архіву після відпрацьовування режимів 6 і 3 (архів згрупований).

```
Уміст файлу FileAdd:
05  212233  шт.    38.45    2800   2500   2810   2356
14  899444  кг     55.15     650    700    660    720
03  456643  шт.   435.00     180    180    182    166
```

```
        АРХІВ ВІДОМОСТЕЙ ПРО ПРОДУКЦИЮ, ЩО ВИПУСКАЄТЬСЯ
-----------------------------------------------------------------
: N  : N  :  Код :Один.:       :  План випуску :  Факт.випуск  :
:п/п :цеху:виробу:вим. : Ціна  :-----------------------------:
:    :    :      :     :       :   1   :   2   :   1   :   2   :
```

1.	1	320011	шт.	221.50	200	180	180	260
1.	5	212233	шт.	38.45	2800	2500	2810	2356
2.	1	320011	шт.	221.50	200	180	180	260
3.	1	324531	шт.	35.14	500	600	510	610
4.	1	337288	шт.	665.70	200	250	220	222
5.	1	338666	шт.	12.10	1900	2500	1990	2110
6.	14	385005	шт.	280.75	110	120	95	98
7.	14	385571	шт.	444.40	150	160	155	159
8.	14	385629	шт.	188.25	250	250	260	252
9.	14	386388	шт.	82.14	560	700	577	655
.....								
25.	5	621412	шт.	53.14	440	460	442	434
26.	5	622277	шт.	28.60	1800	2000	1810	1856
27.	5	623462	шт.	35.30	3200	2900	3060	3070
28.	14	759003	кг	54.70	660	600	630	650
29.	14	769344	кг	88.80	440	450	460	470
30.	14	780088	кг	95.50	740	750	730	760
31.	14	899444	кг	55.15	650	700	660	720

5. Таблиці обробки архіву (режим 9).

ВІДОМОСТІ ПРО ВИКОНАННЯ ПЛАНУ ПО ЦЕХУ 14

	Півріччя		За рік
	1	2	
План випуску	342687.40	363119.50	705806.90
Фактичний випуск	343157.03	355680.80	698837.83
Відсоток виконання	100.14	97.95	99.01

СПИСОК ВИРОБІВ, ПО ЯКИМ НЕ ВИКОНАНИЙ РІЧНИЙ ПЛАН ВИРОБНИЦТВА

N п/п:	Код виробу	Найменування виробу	Виконання плану, %
1	337288	Ресивер	98.22
2	338666	Кільце обтискне	93.18
3	385005	Корпус двигуна	83.91
4	386388	Маховик	97.78
5	386410	Заготівля вала	81.00
6	410410	Різець прохідної	84.31
7	521682	Пускач магнітний	88.18
8	521695	Реле теплове	96.22
9	530510	Трансформатор струму	98.66
10	620405	Штамп прес-форми	96.06
11	621412	Панель монтажна	97.33

---

8. Рекомендації по виконанню контрольної робот

Для конкретизації тексту рекомендації, що нижче приводяться, орієнтовані на розробку описаної вище програми Kontr4.

1. На першому етапі розробки програми доцільно реалізувати процедури, призначені для читання основного вхідного файлу FileInput і формування архівного файлу FileOut. Для цього потрібно в мінімальному обсязі сформувати модулі DesUnit, BasUnit, WorkUnit і основну програму Kontr4. Формування модуля InpUnit, призначеного для перевірки коректності вихідних даних, відкласти на кінцевий етап розробки.

У модулі DesUnit записати оголошення констант MaxKodif, Enter, PressKey, типу ProductType, змінних np, KeyRegime, Product, оголошення файлів.

У модулі BasUnit розмістити основні сервісні процедури й функції: WaitEnter, PrintString, WritelnString, PrintKeyAndWaitEnter, Space, NotSpace, GetNumber.

У модуль WorkUnit записати процедури ReadProduct і CreateArchive.

Основну програму Kontr4 сформувати в повному вигляді, але в операторі **Case** записати звернення лише до процедури CreateArchive, для інших констант селектора встановити порожнє значення.

У файлі FileInput записати 3-4 рядка інформації, що відповідає типу ProductType.

У процедуру CreateArchive після звертання до процедури ReadProduct тимчасово вставити оператор Writeln для виводу на екран умісту прочитаного запису. Після перевірки правильності читання файлу FileInput цей оператор вилучити.

2. У модуль WorkUnit додати процедуру друку архіву PrintArhive, в основну програму - звертання до цієї процедури, у модуль BasUnit - використовувану при цьому процедуру UsesPrinter, в DesUnit - відповідні оголошення змінних.

У файл FileInput додати 25-30 рядків інформації.

Перевірити й налагодити роботу режиму друку архіву.

3. Виконати аналогічні дії стосовно кодифікатора виробів.

4. У модуль DesUnit додати оголошення типів і змінних, необхідних для формування черги виробів, у модуль WorkUnit - процедури CreateDynProduct, WriteFileProduct, SortArchive.

Виконати налагодження режиму сортування компонентів архіву.

5. Послідовно реалізувати режими додавання компонентів в архів, вилучення і зміни компонента архіву, обробки архіву.

6. Сформувати модуль InpUnit, послідовно виконати всі перевірки, створюючи у вхідних файлах відповідні помилкові ситуації.

Тут слід зазначити, що з модуля InpUnit, описаного в програмі Kontr4, понад половину тексту може бути використано студентами при розробці інших завдань. Це стосується перевірки наявності текстового файлу, вилучення порожніх рядків у файлі, перевірки форматів, діапазонів та ін..

Раніше вже відзначалося, що оголошення констант, типів і змінних у програмі повинне бути змістовним і відповідати умові завдання. Для цього доцільно використовувати англійські слова, але не копіювати не підходящі за змістом імена із програми Kontr4. Наприклад, було б некоректно для типу запису варіанта 1, у якому мова йде про розклад руху літаків, використовувати ім'я ProductType (Product - по-англійському "виріб"). Тут тип запису можна було б оголосити в такий спосіб:

```
Type
   TimeType = record
      Hour : 0..23;           { години }
      Minute : 1..59;         { хвилини }
   end;
   PriceType = record
      Griv : word;            { гривні }
      Kop : byte              { копійки }
   end;
   TimeTableType = record
      Trip : 1000..9999;      { номер рейсу }
      EndPort : string[20];   { аеропорт призначення }
      Distance : word;        { відстань }
      AdultTicket,            { ціна дорослого квитка }
      ChildsTicket            { ціна дитячого квитка }
                 : PriceType;
      DepartureTime,          { час відправлення }
      ArrivalTime             { час прибуття }
                 : TimeType;
   end;
```

ВАРІАНТИ ЗАВДАНЬ

Загальні зауваження

У кожній програмі, що реалізує відповідний варіант завдання по контрольній роботі № 4, обов'язковими режимами роботи є:

- контроль коректності вхідних даних;
- створення типізованого файлу, що містить записи оброблюваного архіву;
- угруповання архіву по якій-небудь ознаці;
- друк архіву.

Формовані списки, що одержуються при обробці архіву, також повинні бути згруповані по певній ознаці (по зростанню шифру або коду, за алфавітом прізвищ і т.п.).

Обов'язковими операціями при зміні архіву є доповнення архіву новими записами і вилучення з архіву існуючих записів. Змінювані реквізити вказуються в завданні.

Кількість оброблюваних записів залежить від умови завдання, але, як правило, повинне бути не менш 20.

Імена змінних у програмі повинні бути змістовними.

---

1. У розкладі руху літаків з аеропорту м.Донецька зазначені наступні відомості: номер рейсу (4 цифри); аеропорт призначення; відстань у км; вартість квитка (дорослий і дитячий); час у годинах і хвилинах (відправлення і прибуття). Сформувати таблицю рейсів для заданого аеропорту призначення, а також таблицю відомостей про три рейси з максимальною тривалістю польоту. Коректовані реквізити: вартість квитка, час відправлення і прибуття.

---

2. У магазині є наступні записи про товари: назва, одиниця виміру, код (10 цифр), ціна (грн і коп), кількість прибулого, проданого і товару, що залишився, за поточну добу. Скласти таблицю проданого товару, згрупувавши її по зменшенню загальної вартості продажу, і таблицю п'яти товарів, найбільших по кількості залишку. Коректовані реквізити: ціна і кількості.

---

3. На заводі радіоелектроніки випускають звуковідтворюючу техніку і є наступні дані: назва приладу, призначення (магнітофон, програвач і т.п.), рік створення, вартість, гарантійний строк експлуатації. Необхідно скласти список магнітофонів, розроблених у заданому році, а також список приладів, гарантійний строк яких більше трьох років. Коректовані реквізити: вартість, строк гарантії.

---

4. При проведении підсумків референдуму в м.Донецьку від різних районов була отримана інформація: назва району, загальна кількість жителів, що мають право голосу, число жителів, принявших участь в референдумі, число жителів, що голосували "Так" і голосували "Ні". Необхідно скласти список районов, які більшістю голосів проголосовали "Так", і список районів, у яких голосувало менше 50% жителів, що мають право голосу. Коректовані реквізити: дані про число жителів (4 графи).

---

5. На станції технічного обслуговування автомобілів ведеться облік автомобілів, які пройшли капітальний ремонт, за наступними даними: марка машини, реєстраційний номер (три букви і п'ять цифр), пробіг (у км) після попереднього ремонту, рік випуску машини. Необхідно скласти список автомобілів, що мають пробіг більше 100 000 км, а також список п'яти самих нових машин, які пройшли ремонт. Коректовані реквізити: пробіг, рік випуску.

---

6. В аптеці ведеться облік лекарських засобів. Є наступні дані: назва ліків, ціна упакування, кількість упакувань, рік і місяць випуску, строк зберігання (в роках і місяцях). Віддрукувати список ліківв, не придатних до вживання на заданий рік, і п'ять самих дорогих ліків. Коректовані реквізити: ціна, кількість, строк зберігання.

---

7. У заводському цеху ведеться журнал витрат матеріалів за наступними даними: назва матеріалу, шифр (6 цифр), одиниця виміру (кг, шт. і т.п.), витрата за добу, наявна кількість у цеху. Необхідно вивести список матеріалів, які закінчаться через задану кількість днів, а також список п'яти найменш витратних матеріалів (в абсолютному виразі і в відсотках від наявної кількості). Коректовані реквізити: витрата, кількість.

---

8. За матеріально відповідальною особою значаться матеріальні цінності, які записані в журналі: назва предмета, інвентарний номер (4 цифри), кількість, рік придбання, строк служби (роки). Необхідно вивести список предметів, що підлягають списанню в заданий рік, а також список п'яти самих довгострочних предметів. Коректовані реквізити: кількість, строк служби.

---

9. На АТС ведеться облік міжміскіх розмов абонентів за наступними даними: прізвище й ініціали абонента, домашня адреса (вулиця, будинок, квартира), номер телефону (6 або 7 цифр), кількість і сума міжміських телефонних розмов за місяць (у грн і коп). Необхідно вивести список з 10-ти абонентів, що найбільш широко використовують міжміську мережу, а також по алфавіту список абонентів, що мають шестизначний телефонний номер. Коректовані реквізити: кількість, сума.

---

10. У заводському цеху ведеться облік робочого часу. В журнал заносяться наступні дані: прізвище і ініціали робітника, посада, загальна кількість робочих годин у тижні, кількість відпрацьованих годин, кількість годин, що пропущені через хворобу, кількість годин прогулів. Необхідно вивести список 10-ти робітників з найбільшою кількістю прогулів, а також визначити процент використання робочого часу в середньому по цеху. Коректовані реквізити: кількості відпрацьованих і пропущених годин.

---

11. Диспетчер автовокзалу відзначає автобуси, які прибули на вокзал та які виїхали з вокзалу. Журнал ведеться за наступними даними: номер рейса, номер автобуса (три букви і п'ять цифр), прізвище й ініціали водія, час відправлення (години, хвилини), час прибуття (години, хвилини). Скласти список автобусів, що перебувають у шляху на заданий час. Визначити також три автобуси з найбільшим проміжком часу між відправленням і прибуттям (години, хвилини). Коректовані реквізити: час відправлення і прибуття.

---

12. Турнірна таблиця кубка власників кубків по футболу містить наступні відомості: країна, назва команди, прізвища тренера і капітана, кількості перемог, поразок і нічиїх. Віддрукувати список трьох команд-призерів, а також відомості про команди, які мають більше поразок, ніж перемог. Коректовані реквізити: результати ігор.

---

13. У книзі заявок житлового ремонтного управління містяться відомості про заявки: прізвище і ініціали заявника, його адреса (вулиця, будинок, квартира), тип ремонту (малий, середній, великий), дати заявки і планового строку виконання ремонту, фактичний строк виконання. Віддрукувати список очередників, у яких ремонт виконаний із запізненням у три і більше місяців, а також відомості про трьох першочергових на малий ремонт. Коректовані реквізити: дати.

---

14. У годинниковій майстерні є відомості про проведення ремонтів цього року: прізвище і ініціали замовника, його адреса (вулиця, будинок, квартира), марка годинника, вартість ремонту, дата надходження в ремонт, плановий строк виконання (кількість днів), дата фактичного закінчення ремонту. Передбачити аналіз і друк відомостей про затримки виконання ремонту в заданому місяці. Визначити максимальну і середню затримки. Коректовані реквізити: дати.

---

15. У довідковому бюро міста є наступні відомості: прізвище і ініціали, дата народження, адрес (вулиця, будинок, квартира), домашній телефон (при його відсутності записувати символ «тире»). Віддрукувати відомості про громадян із заданим прізвищем і громадян, які старші заданої кількості років.

Визначити трьох найстарших жителів міста. Коректовані реквізити: адреса й телефон.

---

16. У житловому управлінні є наступні відомості по проживаючим громадянам: адреса (вулиця, номер будинку, номер квартири), прізвище і ініціали квартиронаймача, кількість проживаючих, дата переїзду по даній адресі. Віддрукувати список квартиронаймачів, у яких є зайва площа (норма на людину 13 кв.м). Визначити трьох квартиронаймачів, які займають мінімальну площу в розрахунку на одну людину. Коректовані реквізити: кількість проживаючих і дата переїзду.

---

17. В особистих картках робітників підприємства записані наступні відомості: табельний номер (5 цифр), номер цеху, прізвище і ініціали, дата народження, професія, рік прийому на роботу, розряд. Обрахувати для заданого цеху загальну кількість робітників і кількість робітників, що мають відповідно розряди 3, 4, 5, 6 (абсолютні і відносні значення в % до загальної кількості робітників у цеху). Визначити номер цеху, що має в середньому наймолодший склад робітників. Коректовані реквізити: номер цеху, табельний номер і розряд.

---

18. Дані про студентів вузу містять у собі наступні відомості: прізвище і ініціали, дата народження, номер студбілету (6 цифр), факультет (до 5 букв), група, назва дисципліни й оцінки екзаменаційної сесії (4 предмети). Для заданої дисципліни визначити кількість оцінок 5, 4, 3 і 2 (абсолютні і відносні значення в %). Віддрукувати список відмінників. Коректовані реквізити: оцінки.

---

19. По кожному з виробів, що випускаються заводом, є наступні відомості: номер цеху, код виробу (7 цифр), найменування виробу, кількість виробів, собівартість, оптова ціна. Потрібно для кожного цеху визначити: сумарні витрати, сумарний дохід, коефіцієнт ефективності. Визначити в цілому по заводу три самих масових вироби (по сумарній собівартості). Коректовані реквізити: кількість, собівартість, ціна.

---

20. Є відомості про наявність ПЕОМ на кафедрах вузу: скорочена назва кафедри, тип ПЕОМ, параметри (тактова частота, ємність оперативної пам'яті, ємність вінчестера), вартість, кількість, використання (навчальна робота, НДР), рік випуску. Віддрукувати список навчальних ПЕОМ із тактовою частотою не менш 200 Мгц і "віком" не більше 5 років, а також визначити їхню загальну вартість. Указати список трьох кафедр, на яких установлено найбільшу кількість ПЕОМ для НДР. Коректовані реквізити: вартість, кількість, використання.

---

21. На поштамті є наступні дані про передплатні періодичні видання (газети і журнали): номер квитанції (6 цифр), прізвище і ініціали передплатника, домашня адреса (номер поштового відділення, вулиця, будинок, квартира), індекс видання (5 цифр), найменування видання, номер першого місяця підписки, кількість місяців підписки, вартість підписки. Віддрукувати дві групи відомостей:

а) по виданням - індекс, найменування, кількість передплатників, сума підписки, середня кількість місяців підписки;

б) по поштовим відділенням - кількість видань, кількість підписок, сума підписок.

Коректовані реквізити: номер і кількість місяців підписки, вартість підписки.

---

22. По кожній з лабораторій вузу є наступні дані: номер лабораторії (3 цифри), назва лабораторії, скорочена назва кафедри, назва кафедри, тип лабораторії (навчальна або наукова), її площа, кількість робочих місць. Для заданої кафедри сформувати таблицю: шифр і назва кафедри, кількість лабораторій (загальна, навчальних, наукових), їхня площа, кількість робочих місць. Дати відомості про три кафедри, що мають максимальну загальну площу лабораторій. Коректовані реквізити: площа і кількість робочих місць.

---

23. Турнірна таблиця чемпіонату країни по футболу містить наступні відомості: номер команди (дві цифри), найменування команди, місто, прізвище і ініціали капітана, кількість проведених ігор, результати ігор (кількості перемог, поразок, нічиїх), кількості забитих і пропущених м'ячів. Потрібно підрахувати кількість очків для заданої команди і віддрукувати дані про команди в порядку їхнього положення в турнірній таблиці. Визначити дві команди із кращою різницею забитих і пропущених м'ячів. Коректовані реквізити: всі числа.

---

24. Дані записи про витрату електроенергії на заводах області. Структура запису: номер заводу, найменування заводу, прізвище і ініціали директора, прізвище і ініціали головного енергетика, витрата електроенергії в тис. квт-г (плановий і фактичний). Для кожного заводу підрахувати розмір відхилення фактичної витрати від планової (абсолютне значення і відносне в % з урахуванням знака відхилення). Віддрукувати відомості про два заводи з максимальним відносним значенням економії електроенергії. Коректовані реквізити: планові і фактичні витрати електроенергії.

---

25. Інформація про продаж товарів має наступну структуру: номер магазина, номер секції, номер чека (буква і 4 цифри), найменування товару, артикул товару (6 цифр), ціна товару (грн., коп.), кількість товару, дата продажу. Потрібно визначити товарообіг (загальну суму виторгу) по заданому

магазині. Вибрати дві секції з максимальним товарообігом. Коректовані реквізити: ціна і кількість товару.

---

26. Для обробки на ЕОМ надійшли відомості про виробництво деталей за минулий тиждень (тиждень вважати шестиденним). Структура запису: шифр виробу (5 цифр), дата, номер цеху, номер ділянки, табельний номер, код операції, розряд роботи, кількості виготовлених і прийнятих деталей. Для заданого цеху і заданого дня (дати) визначити кількість бракованих деталей. Визначити також день тижня, у який кількість бракованих деталей було випущено найбільше. Коректовані реквізити: розряд і кількості деталей.

---

27. Шаховий чемпіонат проводиться по круговій системі. У чемпіонаті беруть участь n спортсменів ($6 \le n \le 10$). За результатами кожної гри складається відомість за формою: номера першого і другого спортсменів, їх прізвища, результат гри (1:0, 0:1, 0.5:0.5), час ігор першого і другого спортсменів (години, хвилини). Потрібно для кожного спортсмена віддрукувати: номер і прізвище, кількості перемог, нічиїх і поразок, кількість набраних очків, загальний час ігор.

Коректовані реквізити: результати і час.

---

28. На кожного із спортсменів, які заявлені на змагання з легкої атлетики, представлена картка з наступними даними: реєстраційний номер (три цифри), місто, прізвище і ініціали, вік (років), ріст (см), код виду змагань (не більше чотирьох видів). Потрібно віддрукувати для заданого міста: кількість спортсменів, у тому числі по інтервалах віку (до 18 років, від 18 до 20 років, від 20 до 25 років, понад 25 років), середній вік спортсменів (у дробовій частині числа - одна цифра), кількість заявлених видів змагань. Віддрукувати відомості про трьох самих рослих спортсменів. Коректовані реквізити: вік, ріст, код.

---

29. По кожному з автомобілів, наявних на автобазі, представлені наступні відомості: реєстраційний номер (3 букви і чотири цифри), тип, заводський номер, рік випуску, кількість пройдених кілометрів, оцінка технічного стану (по чотирьохбальній системі). По кожному типу автомобіля визначити середню оцінку технічного стану (у дробовій частині числа - одна цифра). Віддрукувати відомості про три автомобіля з максимальним пробігом. Коректовані реквізити: кількість км, оцінка технічного стану.

---

30. На АТС для кожної міжміської розмови складається картка з наступними даними: телефон абонента, його прізвище і ініціали, домашня адреса (вулиця, будинок, квартира), замовлене місто, відстань у км, телефон замовлення, дата і час розмови (у хв). Віддрукувати відомості для оплати переговорів, вважаючи, що вартість одної хвилини становить: 10 коп. - до 200

км, 20 коп. - до 500 км, 30 коп. - понад 500 км. Визначити три міста, для яких час переговорів максимальний. Коректовані реквізити: відстань, дата, час.

---

31. У фірмовому магазині радіоелектроніки ведеться облік продажу телевізорів: дата продажу, номер касового чека, тип телевізора, вартість, країна-виготовлювач, наявність рекламації покупця, відмітка про обмін телевізора або вартість гарантійного ремонту (при наявності рекламації). Віддрукувати список телевізорів, виготовлених у Гонконгу, Сінгапурі і Малайзії, на яких надійшли рекламації протягом року після їх продажу, а також список трьох типів телевізорів з найбільшою сумою виторгу. Коректовані реквізити: вартість, рекламація, обмін, гарантійний ремонт.

---

32. Є списки мешканців будинку, складені в наступному порядку: номер квартири, прізвище і ініціали квартиронаймача, кількість членів родини, кількість дітей до 16 років, число кімнат, загальна площа. Сформувати список багатодітних родин (більше трьох дітей) і список перенаселених квартир при нормі 13 кв.м/чол. Коректовані реквізити: кількості членів родини і дітей.

---

33. На заочну шкільну олімпіаду надійшли рішення опублікованих завдань. Після їхньої перевірки створена база даних з інформацією про учасників: прізвище і ініціали, місто, предмет (інформатика, фізика, математика, хімія), бали. Якщо той самий школяр бере участь в олімпіадах по декількох предметах, то по кожному предмету формується окремий запис. Роздрукувати дані про три переможця по кожному із предметів. Визначити, яке місто прийняло найбільш активну участь в олімпіаді. Коректовані реквізити: бали.

---

34. В бухгалтерії підприємства ведеться облік зарплати за такою формою: рік, місяць, відділ (дві або три букви), прізвище і ініціали працівника, зарплата, премія, сума до видачі (прийняти податок 20 % від зарплати+премії). Скласти по кожному відділу список працівників, яким видана сума, що перевищує середню по відділу. Визначити три відділи із загальною максимальною виплаченою сумою. Коректовані реквізити: зарплата, премія.

---

35. На залізничному вокзалі м.Донецька є інформація про продані квитки: номер поїзда, станція призначення, дата і час відправлення, кількість годин у шляху проходження, станція, на яку проданий квиток, і вартість квитка (грн. і коп.). Визначити список поїздів, що відбувають із Донецька з 8 до 12 годин. Визначити також три номери поїздів, на які продано найбільше квитків у вартісному вираженні. Коректовані реквізити: дата і час відправлення, вартість квитка.

---

36. На складі ведеться облік товарів за такою формою: номер складського корпуса, номер секції, шифр товару (три букви і три цифри), ціна, кількість, мінімальна норма. Скласти список товарів, кількість яких нижче мінімальної норми. Визначити також корпус, у якому вартість товарів найбільша. Коректовані реквізити: ціна, кількість, мінімальна норма.

---

37. По мережі кабельного телебачення є наступна інформація: вулиця, номер будинку, номер квартири, прізвище і ініціали, тип телевізора, вартість обслуговування на місяць, оплата за кожний з попередніх шести місяців ("Так", "Ні"). Сформувати таблицю отриманої орендної плати по кожному будинку за 6 місяців, а також список боржників за 2 і більше месяця. Коректовані реквізити: вартість обслуговування, дані про оплату.

---

38. У записній книжці є наступні відомості: прізвище і ініціали, адреса (місто, вулиця, будинок, квартира), телефон, дата народження. Віддрукувати список іменинників, чий день народження наступить не пізніше ніж через місяць після заданої дати, а також трьох найбільш молодих іменинників, що не мають телефону. Коректовані реквізити: телефон, дата народження.

---

39. У відділі збуту заводу є наступна інформація: скорочена назва замовника, шифр товару, найменування товару, ціна, кількість, плановий і фактичний строки поставки (рік, місяць, день). Сформувати список замовлень, по яких затримка поставки перевищує 10 днів, а також список замовників із вказівкою в порядку зменшення їхньої загальної суми замовлення. Коректовані реквізити: ціна, кількість, строки поставки.

---

40. По заводу є дані про виробництво за рік: номер бригади, код виробу, собівартість, загальна кількість, кількість браку. Один виріб можуть робити різні бригади (але в повному обсязі, без кооперації). Скласти узагальнені списки для бригад і для виробів. Визначити бригаду, що має найменшу відносну кількість браку (у вартісному вираженні). Коректовані реквізити: собівартість і кількості.

---

41. На АТС є заявки на установку телефону: адреса (вулиця, будинок, квартира), прізвище і ініціали заявника, дата заявки (рік, місяць, день), відомості про каблирування будинку ("Так","Ні"). Скласти список клієнтів, що подали заявки в останні 30 днів стосовно заданої дати. Віддрукувати відомості про п'ять першочергових, що проживають у каблируваних будинках. Коректовані реквізити: дата заявки, відомості про каблирування.

---

42. У поштовім відділенні є відомості про передплатників: адреса (вулиця, будинок, квартира), прізвище і ініціали, індекс і назва видання, ціна за місяць, початок і кінець підписки в місяцях. Віддрукувати список

передплатників газети "Вечірній Донецьк" на строк не менше 6 місяців, а також таблицю видань із вказівкою кількості і суми підписки. Коректовані реквізити: ціна, строк підписки.

---

43. Інформація про продаж товарів у магазині має такий вигляд: номер секції, найменування товару, артикул (7 цифр), ціна, кількість проданих одиниць, дата продажу (рік, місяць, день). Віддрукувати загальну таблицю проданих товарів, а також список трьох товарів, загальний обсяг продажу яких за 7 днів перед заданою датою максимальний. Коректовані реквізити: ціна, кількість, дата продажу.

---

44. Результати участі гравців футбольної команди в матчах чемпіонату записані в наступній таблиці: прізвище і ініціали, рік народження, рік включення до основного складу, кількість зіграних матчів, кількість забитих м'ячів. Віддрукувати таблицю гравців молодших 25 років і граючих в основному складі не менше двох років, а також список трьох найбільш результативних гравців (по відношенню «кількість м'ячів/кількість матчів»). Коректовані реквізити: кількості матчів і забитих м'ячів.

---

45. Таблиця відбірних ігор чемпіонату світу з футболу містить наступну інформацію: номер відбірної групи, країна, прізвища тренера і капітана, кількості очків, забитих і пропущених м'ячів. Віддрукувати відомості по командам заданої групи, а також список країн, що зайняли перше місце в кожній групі. Коректовані реквізити: очки, кількість м'ячів.

---

46. На метеостанції ведеться облік погодних умов протягом місяця. В журнал щодня вносяться наступні дані: число, температура повітря вдень і вночі, тиск повітря, процент вмісту кисня, радиоактивний фон, тривалість світлового дня. Необхідно вивести список погодних умов тих днів, у які радиоактивний фон перевищував задану величину, а також визначити середню денну й нічну температури за місяць. Коректовані реквізити: всі, крім числа місяця.

---

47. Річний звіт швейної фабрики про випуск продукції містить наступні відомості: номер цеху, найменування виробу, артикул (7 цифр), обсяг випуску плановий і фактичний по товарах I, II і III сорту. Частина виробів випускається тільки I або тільки I і II сортів. Віддрукувати таблицю виробів I сорту, по яких виконаний план їхнього випуску, а також список трьох цехів, які в найбільшій мірі не виконали план. Коректовані реквізити: обсяги випуску продукції.

---

48. На Уімблдонському турнірі є наступні відомості про тенісистів: ім'я і прізвище, країна, дата народження, рейтинг, кількість виступів в Уімблдоні. Скласти список тенісистів молодших 26 років і не менше трьох разів

виступавших в Уімблдоні, а також список п'яти тенісистів з максимальним рейтингом. Коректовані реквізити: рейтинг, кількість виступів.

---

49. У касах попереднього продажу залізничних квитків є наступні відомості: номер поїзда, станція призначення, дата відправлення, кількості місць і проданих квитків (м'яких, купейних, плацкартних і загальних). У деяких поїздах м'які або загальні місця можуть бути відсутніми. Для заданої дати скласти список поїздів, для яких кількість проданих квитків перевищує 90 % їхньої загальної кількості, а також відомості про три поїзди, у яких залишилися лише загальні місця. Коректовані реквізити: кількості місць і квитків.

---

50. На іподромі є відомості по скачкам: ім'я коня, його вік, вага, рейтинг, кількість виграшів. Скласти список коней, які молодше 4 років і мають не менше п'яти виграшів, а також відомості про трьох коней з максимальним рейтингом. Коректовані реквізити: вага, рейтинг, кількість виграшів.
