Как организовать самостоятельную работу при обучении программированию

При дистанционном образовании преподаватель переходит из категории «хранилища знаний» в категорию «гида» в информационном пространстве. Основная роль при этом отводится самостоятельной работе студента как инструменту самообучения

При дистанционном обучении программированию наряду с учебными курсами, которые основаны на традиционных концепциях «лектор», «ассистент» и «наставник» [1 - 4], необходимы курсы для самостоятельной работы студента (СРС). Потребность в таких учебных курсах также обусловлена развитием института экстерната в высшем образовании.

Обучение программированию значительно отличается от обучения другим дисциплинам: гуманитарным, социально-экономическим, технологическим и пр. Далее обсуждаются аспекты обучения профессиональному программированию (для студентов направления «Компьютерные науки») в отличие от программирования общеобразовательного.

Во-первых, программирование (т.е. процесс решения задачи на ЭВМ с помощью языков программирования) находится на «стыке ремесла и искусства». В «ремесле» программирование представляет собой обычную инженерную деятельность человека: есть ряд технологических этапов, которые соответствуют этапам жизненного цикла программного изделия (Постановка задачи, разработка проекта и метода решения, построение алгоритма и его тестирование, кодирование алгоритма средствами языка программирования, отладка программы или ее верификация/валидация, документирование изделия, сопровождение программного продукта) и которые необходимо пройти разработчику программы. Если следовать «технологии», то одну и ту же задачу несколько обучаемых должны решить одинаково. Однако в реальной практике большую роль играет «человеческий фактор»: предыдущий опыт программирования, привязанности обучаемого к определенной парадигме программирования, привычный стиль написания кода программы по аналогии со стилем (и даже почерком!) письма на естественном языке или стиле изложения (в диапазоне от небрежного до изящного или витиеватого, от легкого до тяжеловесного в понимании), желание оценить компетентность преподавателя и т.д. Обучение ремеслу происходит гораздо быстрее, т.к. с увеличением опыта программирования значительная часть знаний переходит на уровень навыков. «Искусство» программирования (как и искусство в целом) основано не только на таланте обучаемого «от Бога» или озарении, но и на педантичном освоении теории и практики программирования. Современные информационные технологии обеспечивают «дружественную» пользователю среду разработки программ, выполняя средствами инструментов и генераторов программ основную рутинную работу программирования, и обеспечивают профессиональному пользователю больше простора для совершенствования в «искусстве».

Во-вторых, процесс обучения программированию - непосредственный и интерактивный - выполняется с помощью ЭВМ, соответствующего программного обеспечения и встроенных в него мощных систем контекстной помощи. Обучаемый программированию имеет немедленную «обратную связь» с разрабатываемым им программным изделием, выступая одновременно в ролях постановщика задачи, разработчика и пользователя.

В-третьих, с точки зрения разработчика компьютерных обучающих программ современные информационные технологии позволяют достаточно просто решить проблему разработки и реализации «решателя задач» [5] при обучении программированию: в роли «решателя» достаточно использовать системное и прикладное программное обеспечение.

Многолетний опыт обучения студентов специальности «Программное обеспечение» программированию на языках, основанных на различных парадигмах информатики (алгоритмической, функциональной и логической), позволил сделать вывод о необходимости расширения видов самостоятельной деятельности обучаемых при изучении языков. Обучение искусственным языкам (какими являются языки программирования) в значительной степени напоминает обучение естественным языкам.

Определим структуру учебного материала при дистанционном обучении программированию которая используется для активизации самостоятельной работы студента. Индивидуальное задание

студента по каждой теме предусматривает выполнение последовательности шагов:

1) Входное тестирование по определениям понятий, синтаксису и семантике конструкций языка, особенностей их применения.

2) «Чтение»: Чтение текста программы на языке программирования и запись его семантики на естественном языке, используя терминологию текущей темы.

3) «Диктант»: По тексту на естественном языке написание фрагмента программы на языке программирования с использованием данных (констант, переменных и выражений) изучаемого типа данных: их объявление, инициализация или ввод значений, обработка одним из методов решения задач на ЭВМ, вывод.

4) «Библиотека»: Использование наиболее распространенных функций стандартных библиотек (встроенных функций) для работы с изучаемым типом данных в фрагменте программы на языке программирования.

5) «Понимание»: Анализ фрагмента программы на языке программирования, который содержит типовые ошибки использования изучаемого типа данных и конструкций языка, с целью его верификации и указание перечня обнаруженных ошибок и соответствующих действий по их устранению.

6) «Сочинение»: Разработка программы на языке программирования для решения задачи с использованием изученного типа данных и конструкций языка в соответствии с индивидуальным заданием.

7) Заключительное контрольное тестирование по определениям понятий, синтаксису и семантике конструкций языка, особенностей их применения.

Пример 1: Задание по теме «Работа с символьными и текстовыми данными на языке Си. Функции, определяемые пользователем» (дисциплина «Основы программирования и алгоритмические языки»),

1. Входное тестирование:

- представление символьных данных в оперативной памяти;
- представление строковых констант в оперативной памяти;
- способы представления строковых переменных в оперативной памяти;
- способы описания строк в языке Си;
- операции для работы с символами в Си;
- средства для работы со строками в языке Си;
- определение пользователем функции и ее вызов в языке Си;
- «формальные» и «фактические» аргументы функции;
- назначение прототипа функции в языке Си;
- особенности передачи аргументом функции данных символьного типа и строк в языке Си.

2.Записать фрагмент деклараций программы, соответствующий тексту на естествен-ном языке:

а) объявить переменную c1 символьного типа и инициализировать ее значением ‘+’;

б) объявить строку состоящую из 10 элементов, как литерный массив. Инициализировать ее значением 2.1743- Объявить литерный указатель pc и инициализировать его значением 2.1743. Проиллюстрировать размещение в памяти строк s1 и pc;

в) объявить строку s3 как литерный массив, не указав длину строки, и инициализировать ее значением 2.1743. Проиллюстрировать размещение в памяти строки s3 Определить количество элементов в этом массиве по результатам инициализации;

г) объявить массив s4 из 4 строк как двумерный символьный массив и массив строк s5 как массив литерных указателей без указания количества элементов; инициализировать s4 и s5 значениями if, for, while. Проиллюстрировать размещение в памяти массивов s4 и s5. Определить количество байтов, которое занимают массивы s4 и s5 по результатам инициализации.

3. Записать фрагмент программы, соответствующий тексту на естественная языке:

а) проверить, является ли 2-й символ третьего слова массива s5 строчной буквой. (Задание выполнить с помощью соответствующего макроса библиотеки <ctype.h>);

б) найти в строке s2 первое вхождение символа cl, результат вернуть в переменной pc . (Задание выполнить с помощью соответствующей функции библиотеки <string.h>);

в) объявить прототип функции f1 с 2 аргументами, первый из которых является символом, второй - строкой в виде литерного массива; f1 возвращает результат целого типа:

г) вызвать эту функцию с аргументами «2» и «02.07.1992 - дата»; '

4. *Найти ошибки в программе:*

```
char si[];
char s2="2.07";
if (sl==s2))
char s[4]="27";
char t[4]=27;
char y[3]='k';
char x;
char y[]="a11";
x=isdigit(y);
char * x='a11';
char y[]="12";
x=strcpy(y);
char f(int z, char c);
 {char c1, cr;
  int il; ff=(c1<cr?i1:-l);}
  cr=f(cr, i1, 5);
```

5. *Записать на естественном языке семантику работы фрагмента программы:*

```
#include <ctype.h>
#include <string.h>
#define N 10
 #define M 5
#define L 20
main ()
  {char c1 = '-';
   char 'pc;
   char s5[M][L] = {"For-Next",
    "if-then-else",
    "DO-WHILE",
    "switch-case"};
```

```
 int i, j, pr=l;
 for (i=0; i<M && pr; i++)
if(islower ((int) S5[i][0]))
{pc=strchr(s5[i] ,cl);
 pr=0; }
 }
```

6. *Решить задачу с помощью ЭВМ. Обязательно использовать функции, определенные пользователем.*

В исходном тексте есть изображения двузначных чисел. Переписать этот текст, заменяя найденное число его словесным эквивалентом.

Например, для текста

33 коровы, 33 коровы,

33 коровы – свежая строка

(Песня из кинофильма "Мэри Попинс")

результат будет

тридцать три коровы, тридцать три коровы,

тридцать три коровы – свежая строка

(Песня из кинофильма "Мэри Попинс")

7. *Заключительное тестирование:*

- реализовать на языке Си стандартную функцию srcpy, моделируя строки:
  - литерными массивами;
  - литерными указателями (использовать адресную арифметику);
- реализовать алгоритм подсчета слов текста t, состоящих только из символов в верхнем регистре.
- организовать на языке Си ввод текста S произвольной длины и вывод его по 64 символа в строке;

Описанный подход использован при разработке дистанционного учебного курса для самостоятельной работы по дисциплинам «Основы программирования и алгоритмические языки». Реализация выполнена средствами HTML и JavaScript.

Для контрольного тестирования разработана система генерации тестов для работы в сетевом варианте. Индивидуальное задание на тестирование студента формируется из базы данных тестов случайным образом, причем порядковый номер варианта правильного ответа в случае повторного выбора вопроса в тесте также генерируется случайным образом из базы данных ответов. Реализация выполнена средствами HTML, JavaScript и Perl.

Литература

1. Seagren A. T., Watwood B. Using the computer to deliver global graduate education // Proc. ICDED'96. - Moscow, 1996. - P. 90-93.
2. Loing B. Learning environments: new skills for old issues // Proc. ICDED'96. - Moscow, 1996. - P.235-238.
3. Мельников И. А., Монкус В. В., Тамм Б. Г. Обзор и анализ зарубежных компьютерных обучающих систем в области программирования // Прикладная информатика. – М. 1984.- Вып.15,- С. 131-153.
4. Дацун Н.М. Функціональне і логічне програмування: сучасні підходи до комп'ютерного навчання // Проблеми освіти: Наук.-метод. збірник. - Вип.1,- Київ: Ін-т системних досліджень освіти, 1995. - 10с.
5. Петрушин В. А. Экспертно-обучающие системы. - Киев: Наук думка, 1992. - 196с.

Наталья Дацун

Статья представлена 23.07.2000