Методический опыт организации курсовой работы первокурсников по информатике и программированию

О. Д. Горбенко, e-mail: oleg_dan@mail.ru
О. Ф. Ускова, e-mail: nat-uskova@mail.ru

Воронежский государственный университет

Аннотация. *В работе рассматриваются вопросы организации и методической поддержки нового элемента учебного плана по направлению обучения 02.03.02 – Фундаментальная информатика и информационные технологии. Рассматриваются требования к выполнению курсовой работы, оформлению ее результатов в виде отчета. Приводится пример отчета по курсовой работе.*

Ключевые слова: *учебный план, информатика, программирование, структуры управления, структуры данных, функции.*

Введение

В 2019-2020 учебном году в учебный процесс для студентов-первокурсников, обучающихся по направлению 02.03.02, был введен новый элемент учебного плана – курсовая работа по дисциплине «Информатика и программирование». Цель курсовой работы – закрепить и продемонстрировать знания в области базовых структур данных и структур управления и навыки в разработке программ на языке программирования с использованием функций.

Курсовая работа выполняется самостоятельно во внеаудиторное время. На ее выполнение отводится один месяц – с 1 по 30 апреля. Завершается выполнение курсовой работы представлением отчета, оформленного в соответствии с установленными требованиями, и собеседованием с руководителем курсовой работы.

1. Содержание курсовой работы

Курсовая работа по информатике и программированию состоит в программной реализации индивидуального практического задания с использованием таких языковых средств, как статические и динамические массивы данных, указатели, функция.

Основной результат курсовой работы – три программы (или три части одной программы):

– в первой для решения поставленной задачи используется традиционный подход к объявлению и созданию статических массивов данных;

– во второй применяются языковые инструменты объявления и создания динамических массивов данных (с использованием адресной арифметики);

– в третьей части для реализации одного из многошаговых действий (в любой из предыдущих частей) следует определить и применить функцию.

Завершается выполнение курсовой работы представлением отчета, оформленного в соответствии с установленными кафедрой требованиями. Отчет должен включать следующие шесть частей.

1. Постановка задачи.

В этой части следует разместить полное описание поставленной задачи так, как оно дается в задании.

2. Описание данных и алгоритма решения задачи.

Здесь необходимо указать имена (и типы, к которым они отнесены) всех простых и структурированных данных, а также дать описание алгоритма решения задачи, для чего можно использовать известные средства описания алгоритмов. Такими средствами могут быть блок-схемы [4-5] и язык псевдокода [2-3].

3. Описание структуры программы.

Отдельные части программы на языке программирования строятся на основе разработанного и описанного в предыдущей части алгоритма с указанием, какой части алгоритма соответствует та или иная часть программы.

4. Результаты тестирования.

Вначале приводятся результаты выполнения разработанной программы для нескольких случаев входных данных, когда достоверность результата очевидна. Затем приводятся результаты выполнения программы для более сложных случаев ввода исходных данных.

5. Список использованных источников.

Здесь перечисляются источники (литература на бумажных носителях или источники в электронной форме, найденные в сети Интернет), которые были использованы при выполнении курсовой работы. В этом списке могут быть учебники, монографии, методические указания или рекомендации, web-страницы, содержащие материал, оказавшийся полезным при выполнении курсовой работы [1-6].

6. Приложение.

В этой части отчета по курсовой работе размещается полный текст программы и результаты ее работы.

Разработанный в настоящее время банк заданий для курсовой работы первокурсников, обучающихся по направлению 02.03.02, содержит 80 различных задач [1-6]. Ниже приводятся примеры таких задач.

2. Примеры постановки задачи

1. С клавиатуры вводится информация об итогах последней экзаменационной сессии. Эта информация включает в себя:

1) целое число *n* – количество студентов;

2) *n* объединенных в структуру данных:

<имя> <фамилия> <оценка> <оценка> <оценка> <оценка> ,

где <*имя*>, <*фамилия*> — символьные строки, содержащие не более 20 символов, <*оценка*> — десятичная цифра из диапазона ‘2’..’5’.

Требуется сформировать массив структур, в котором каждый элемент массива содержит фамилию студента и его средний балл, причем элементы массива располагаются в порядке возрастания средних баллов, и вывести этот массив на экран. Вывести также фамилии и оценки студентов, имеющих наибольшее количество отличных оценок. Для подсчета среднего балла определить и использовать функцию.

Технические требования.

В основной памяти исходные данные также следует хранить в виде массива структур.

Не допускается использование дополнительных массивов.

2. Сформировать квадратную целочисленную матрицу *A* из *m* строк, все элементы которой различны. Сформировать также массив *B* из *m* целых чисел. Требуется найти скалярное произведение вектора *B* на строку матрицы *A*, в которой расположен максимальный элемент матрицы, и удалить эту строку из матрицы *A*, сдвинув следующие за ней строки вверх. Для вычисления скалярного произведения определить и использовать функцию.

3. Оформление отчета

Приведем в качестве примера фрагмент оформленного отчета по курсовой работе и текст разработанной программы на языке С++.

/* Николай Псевдокодов, студ. 1 курса 261 группы, задание №865

С клавиатуры вводится информация об итогах последней экзаменационной сессии. Эта информация включает в себя:

1) целое число *n* – количество студентов; 2) *n* объединенных в структуру данных:

<имя> <фамилия> <оценка> <оценка> <оценка> <оценка> ,

где *<имя>, <фамилия>* — символьные строки, содержащие не более 20 символов, *<оценка>* — десятичная цифра из диапазона ‘2’..’5’.

Требуется сформировать массив структур, в котором каждый элемент массива содержит фамилию студента и его средний балл, а элементы располагаются в обратном лексикографическом (алфавитном) порядке фамилий, и вывести массив на экран. Вывести также фамилии и оценки студентов, имеющих наибольшее количество отличных оценок.

Технические требования.

В основной памяти входные данные также следует хранить в виде массива структур.

Не допускается использование дополнительных массивов.

```
*/
#include <iostream>
#include <string>
#include <windows.h>

/* Входная структура студентов. */
struct student_input
{
      char first_name[20]; // Имя
      char second_name[20]; // Фамилия
      short marks[4]; // Оценки
      int best_mark_count; // Количество пятерок
};

/* Выходная структура студентов. */
struct student_output
{
      char second_name[20]; // Фамилия
      double average_mark; // Средний балл
};

/* Возвращает число студентов, считанное из входного потока.
Проверяет данные на корректность. */
int readCount();

/* Выполняет преобразование данных в информацию о студенте.
Возвращает true, если преобразование успешно.*/
bool parse_student(const int field_index, student_input&
student, std::string& data);
/* Считывает студента из входного потока. Возвращает true,
если считывание успешно.*/
bool readStudent(student_input& student);

/* Возвращает true, если b.second_name > a.second_name. */
bool comparer(student_output& a, student_output& b);

/* Сортирует массив студентов, используя comparer. */
```

```
void sort(student_output* arr, const int length, bool
(*comparer) (student_output& a, student_output& b));

int main()
{
 using namespace std;
 SetConsoleCP(1251);
 SetConsoleOutputCP(1251);

 int student_count;
 while (true)
  {
    cout << "Введите число студентов: ";
    student_count = readCount();
    if (student_count > 0) break;
    cout << "Не удалось распознать число студентов. Введите
ненулевое положительное целое число.\r\n";
   }
  student_input* students = new student_input[student_count];
  int i = 0;
  while (i < student_count)
   {
     cout << "Студент №" << i + 1 << " : ";
     if (readStudent(students[i]))
          ++i;
     else
         cout << "Не удалось распознать студента. Проверьте
соответствие: <имя> <фамилия> <оценка> <оценка> <оценка>
<оценка>.\r\n";
    }
   int max_best_marks = 0;
   student_output* result = new student_output[student_count];
   for (i = 0; i < student_count; ++i)
{
     strcpy_s(result[i].second_name, students[i].second_name);
     int mark_summ = 0;
     for (int j = 0; j < 4; ++j)
      mark_summ += students[i].marks[j];
     result[i].average_mark = double(mark_summ) / 4;
     if (max_best_marks < students[i].best_mark_count)
      max_best_marks = students[i].best_mark_count;
    }
    sort(result, student_count, comparer);
    cout << endl;
    cout << "Фамилии и средние баллы студентов (в обратном
алфавитном порядке):\r\n";
    for (i = 0; i < student_count; ++i)
      cout << result[i].second_name << " " <<
result[i].average_mark << endl;
    cout << endl;
    if (max_best_marks > 0)
    {
```

```
      cout << "Студенты с самым большим количеством
пятерок:\r\n";
      for (i = 0; i < student_count; ++i)
     {
       if (students[i].best_mark_count < max_best_marks)
            continue;
       cout << students[i].second_name;
       for (int j = 0; j < 4; ++j)
            cout << " " << students[i].marks[j];
       cout << endl;
      }
     }
     else
        cout << "Отличников не обнаружено.";
     delete[] students;
     delete[] result;
     cout << "Для продолжения нажмите любую клавишу . . .";
     cin.get();
    return 0;
}
```

Заключение

Предложенная методика организации и структура содержания курсовой работы первокурсников по дисциплине «Информатика и программирование» направлены на закрепление базовых знаний в области программирования и навыков использования языковых средств. На их основе в помощь первокурсникам разработаны методические рекомендации Саму курсовую работу следует рассматривать как пропедевтический этап обучения перед летней учебной практикой.

Список литературы

1. Ускова О. Ф. Начала структурного программирования на языке С++ : задачник-практикум / О. Ф. Ускова, Н. А. Каплиева, О. Д. Горбенко. – Воронеж : Издательский дом ВГУ, 2019. – 261 с.
2. Ускова О. Ф. Задачник–практикум по алгоритмическому языку / О. Ф. Ускова, О. Д. Горбенко. – Воронеж: Издательство Воронежского государственного университета, 1990. – 196 с.
3. Ускова О. Ф. Информатика. Начальный курс : учебное пособие для поступающих на факультет ПММ ВГУ / О. Ф. Ускова, О. Д. Горбенко, Н. А. Каплиева. – Воронеж : Альбион, 2006. – 183 с.
4. Ускова О. Ф. Основы программирования : учебное пособие / О. Ф. Ускова, Н. А. Каплиева – Воронеж : Издательско-полиграфический центр ВГУ, 2010. – 266 с. (Учебник Воронежского государственного университета).
5. Программирование на языке Паскаль : задачник / под ред. Усковой О. Ф. // О. Ф. Ускова, М. В. Бакланов, И. Е. Воронина, О. Д.

Горбенко, Г. Э. Вощинская, Н. В. Огаркова, В. М. Мельников. – СПб. : Питер, 2002 (2003, 2005). – 366 с.

6. Горбенко О. Д. Методические указания к выполнению курсовой работы по информатике и программированию / О. Д. Горбенко, О. Ф. Ускова. – Воронеж : Издательский дом ВГУ, 2020. – 32