1.7. Примеры решения задач обработки массивов

Примеры решения задач обработки массивов

Рассмотрим примеры решения задач с массивами, применяя для организации структуры программы функциональную декомпозицию. Библиотеки функций  оформляют в  отдельные модули и подключают к проекту решения по мере необходимости.

**Задача 6.**  Массив студентов. Известны фамилии и оценки в баллах по 5 дисциплинам каждого из 25 студентов группы. Требуется разработать:

-       функции ввода, вывода  студентов;

-       вычисление среднего балла каждого студента;

-       функцию поиска по ФИО студента и вывод записи о нем;

-       функцию поиска студента с максимальным баллом по заданному предмету;

-       функции сортировки массива студентов по ФИО и среднему баллу;

-       функции чтения/записи массива в текстовый файл, то есть хранение данных на внешнем носителе.

*Решение*

Необходимо реализовать решение задачи на основе массива структур (записей) функциональной декомпозицией.

·        Объявляем тип данных элемента массива – это структура как набор данных разного типа:

**struct Student {**

**char FIO[30];**

**int Ball[5];**

**float SrBall;**

**};**

·        Объявляем статический одномерный массив  с типом данных Student:

**Student G[N];**

·        Описываем функции обработки одномерного массива  со структурированным типом данных элементов.  Алгоритмы будут типовыми операциями над одномерным массивом, но элемент массива нужно обрабатывать, учитывая его структурный тип, то есть по полям структуры. Для переменных структурного типа определена операция присваивания, то есть:

**Student A, B;**

**cout<<"FIO"<<endl; cin.get(A.FIO,30);**

**A.SrBall=0;**

**for (int i=0; i<5; i++) {**

**cin>>A.Ball[i]; A.SrBall+=A.Ball[i];}**

**A.SrBall/=5;**

**B = A;**

**Программа 1.43**Программа частичного[[2]](https://educon2.tyuiu.ru/mod/book/view.php?id=558733&chapterid=1370" \l "_ftn2" \o ") решения задачи 6 «Массив студентов» на основе функциональной декомпозиции.

//----описание типа struct для одного элемента массива--------------------

struct Student {

      char FIO[30];

      int Ball[5];

      float SrBall;

};

const int N=25;  // размер массива - группа

//-----описание функции вывода на экран массива типа Student--------------

void showStudent(Student St[], int size){

     cout<<endl<<"List Studens"<<endl;

     for (int k=0; k<size; k++ ){

            cout<<endl<<St[k].FIO<<' '<<St[k].SrBall;

            for (int i=0; i<5; i++)  cout<<' '<<St[k].Ball[i];

      }

}

//------описание функции добавления в массив   элемента типа Student -----

void addStudent(Student St[], int size){

  char str[30];

  for (int k=0; k<size; k++ ){

        cout<<"FIO"<<endl;cin.get(); cin.get(St[k].FIO,30);cin.get();

        St[k].SrBall=0;

         cout<<"Ball"<<endl;

          for (int i=0; i<5; i++) {

                   cin>>St[k].Ball[i]; St[k].SrBall+=St[k].Ball[i];}

        St[k].SrBall/=5;

      }

}

//--------сортировка массива студентов по среднему баллу------------------

void sortSrBall(Student St[], int size) {

      Student tmp;

      for (int i = 0; i < size - 1; i++)

            for (int j = i + 1; j < size; j++)

                  if (St[i].Srball < St[j].Srball){

                        tmp = St[i]; St[i] = St[j]; St[j]=tmp;

                  }

}

//------------------------------------------------------------------------

int main(){

//-- объявление массива структур

       Student G[N];

       int count=0; // количество  студентов в группе

       cout<<"count "; cin>>count;

//--вызов функций обработки  массива G типа Student

      addStudent(G,count);

      showStudent(G,count);

      sortSrBall(G, count);

      showStudent(G,count);

      cout<<endl;

      system("pause");

      return 0;

}

//------------------------------------------------------------------------