МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

 Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

Отчет по лабораторной работе №5

по дисциплине

Алгоритмы и структуры данных

«Методы сортировки»

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Багиров М. Б.

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сухоруков В.А.

19-ИВТ-3

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2020

**Задача 1**

Имеется N камней веса А1,А2,...,АN.

Необходимо разбить их на две кучи таким образом, чтобы веса куч отличались не более чем в 2 раза. Если этого сделать нельзя, то указать это.

**Входные данные**

Программе через консоль передаются количество камней и их массы.

**Результат работы программы**

Если камни можно разложить по кучам, соответствующим условию, то результат – четыре списка: два с номерами камней и два с их массой. Если разложить нельзя, то выводится соответствующая фраза

**Решение задачи**

Решения заключается в том, что каждый следующий камень кладется в кучу с меньшим текущим весом. При этом в первую кучу надо положить камень максимального веса. Этого достаточно, чтобы гарантировать правильное решение задачи.

По окончании распределения камней по кучам возможны 2 ситуации:

1. Все камни попали во вторую кучу, а ее вес остался меньше половины веса первой кучи. Понятно, что в этом случае камни требуемым образом разбить нельзя, следовательно, решения не существует.
2. Случай 1. не верен. Тогда возможны следующие ситуации.
3. Все камни попали во вторую кучу. В этом случае ясно, что веса куч отличаются не более чем на половину первой кучи, если вес первой кучи больше, или не более чем вес последнего камня, положенного во вторую кучу. В любом из этих случаев требуемое условие выполняется.
4. В первую кучу попали и другие камни. Тогда ясно, что веса куч отличаются не более чем на вес самого тяжелого камня, кроме первого. Следовательно и в этом случае условие задачи выполняется.

**Код библиотеки MyList.h**

#pragma once

#ifndef \_MyList\_H\_

#define \_MyList\_H\_

#include<iostream>

template <typename T> // создание шаблонного класса list

class List {

private:

template <typename T> //создание внутреннего класса для хранения

class Node { //элемента списка

public:

T value; //значение элемента

Node\* Next; //указатель на следующий элемент

//конструктор с параметрами

Node(T val = T(), Node\* n = nullptr) {

value = val;

Next = n;

}

};

int size; //Количество элементов в списке

public: //публичная секция класса

Node<T>\* head; //указатель на первый элемент

List() { // конструктор по умолчанию

head = nullptr;

size = 0;

}

//метод добавления элемента в конец списка

void push\_back(T data) {

//если список пустой, то выделяем память и

//добавляем элемент

if (head == nullptr) {

head = new Node<T>(data);

}

//иначе доходим до последнего элемента,

//выделяем память и добавляем элемент

else {

Node<T>\* cur = head;

while (cur->Next != nullptr) {

cur = cur->Next;

}

cur->Next = new Node <T>(data);

}

size++;

}

//перегрузка оператора, для

//обращения к элементам списка

T& operator[](const int index) {

int counter = 0;

Node <T>\* cur = head;

while (cur != nullptr) {

if (counter == index) {

return cur->value;

}

cur = cur->Next;

counter++;

}

}

int get\_size() {

return size;

}

};

#endif

**Код программы**

#include "MyList.h"

#include<iostream>

using namespace std;

//--Функция для подсчёта массы кучи

template <typename T>

int List\_sum(List<T> list) {

int sum = 0;

for (int i = 0; i < list.get\_size(); i++){

sum = sum + list[i];

}

return sum;

}

//---Перегрузка оператора << для вывода списка

template <typename T>

ostream& operator<<(ostream& os, List<T> list) {

for (int i = 0; i < list.get\_size(); i++) {

os << list[i] << " ";

}

os << endl;

return os;

}

int main(){

setlocale(LC\_ALL, "ru");//Включение русского языка

int N = 0; //Количество камней

int max\_mass=0; //Масса максимального камня

int mass = 0; //Масса текущего в цикле камня

int max\_mass\_index = 0; //Номер камня с максимальной массой

List<float> L\_mass; //Массы камней

List<int> first\_index; //Номера камней первой кучи

List<int> second\_index; //Номера камней второй кучи

List<float>first\_mass; //Массы камней в первой куче

List<float>second\_mass; //Массы камней во второй куче

//---Получение и обработка исходных данных

cout << "Введите количество камней" << endl;

cin >> N;

if (N<=0){

cout << "Некорректное количество камней" << endl;

exit(-1);

}

for (int i = 0; i < N; i++){

cout<<endl << "Введите массу "<<i<<" камня " << endl;

cin >> mass;

if (mass < 0) {

cout << "Некорректная масса камня" << endl;

exit(-1);

}

if (mass>max\_mass){

max\_mass = mass;

max\_mass\_index = i;

}

L\_mass.push\_back(mass);

}

//---Добавление камня с максимальной массой в первую кучу

first\_mass.push\_back(max\_mass);

first\_index.push\_back(max\_mass\_index);

//---Обработка оставшихся камней

for (int i = 1; i < L\_mass.get\_size(); i++){

int sum1 = List\_sum(first\_mass);

int sum2 = List\_sum(second\_mass);

if (sum2 >= sum1) {

first\_mass.push\_back(L\_mass[i]);

first\_index.push\_back(i);

}

else {

second\_mass.push\_back(L\_mass[i]);

second\_index.push\_back(i);

}

}

//-----Обработка полученных масс

int sum1 = List\_sum(first\_mass);

int sum2 = List\_sum(second\_mass);

int max\_sum = sum1;

int min\_sum = sum2;

if (sum2 > max\_mass) {

max\_mass = sum2;

min\_sum = sum1;

}

//--Вывод результата

if (min\_sum\*2<=max\_sum){

cout << "Камни можно разделить на кучи, соглачно условию." << endl

<< "Номера камней в первой куче: " << first\_index

<< "Массы камней в первой куче: " << first\_mass

<< "Масса первой кучи: " << List\_sum(first\_mass)<<endl

<< "Номера камней во второй куче: " << second\_index

<< "Массы камней во второй куче: " << second\_mass

<< "Масса второй кучи: " << List\_sum(second\_mass)<<endl;

}

else {

cout << "Камни нельзя разделить на кучи, соглачно условию."

<< endl;

}

return 0;

}

**Результаты работы программы**







