**Правительство Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего профессионального образования**

**"Национальный исследовательский университет**

**"Высшая школа экономики"**

Московский институт электроники и математики Национального исследовательского университета "Высшая школа экономики"

Департамент прикладной математики

**ОТЧЕТ**

**По лабораторной работе № 2**

**По курсу «Алгоритмизация и программирование»**

|  |  |  |
| --- | --- | --- |
| ФИО студента | Номер группы | Дата |
| Кочарян Давид Рудикович | 203 | 10.10.2021 |
|  |
|  |
|  |

**Москва – 2021 г.**

Вариант 13 (задачи 13 Д,Е)

Изображение выглядит как стол

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Код программы

#include <iostream>  
#include <vector>  
#include <string>  
#include <ctime>  
// программа сама рандомно создает себе базу футболистов  
void shaker (int size, std::string& m );  
  
std:: vector <std::string> get\_fio (int n = 100){  
 std::vector <std:: string> fio (n);  
 int size\_f, size\_n,size\_p;  
 std:: string m;  
 for (int i = 0; i < n; ++i ){// посмотрел сколько обычно выделяют под длину фио, обычно по 50 символов на каждое  
 size\_f = rand() % 50;  
 size\_n = rand() % 50;  
 size\_p = rand() % 50;  
 shaker (size\_f, fio [i]);  
 shaker (size\_n, fio [i]);  
 shaker (size\_p, fio [i]);  
 }  
 return fio;  
}  
  
void shaker (int size, std::string& m) {//Создает слова  
 bool name = 1;  
 if ((m[m.size() - 1] == ' ')or(m.size() == 0)){  
 name = 0;  
 }  
 else{  
 m.push\_back(' ');  
 }  
  
 m.push\_back(char(rand() % 26 + 65));  
 for (int i = 0; i < size - 1; ++i){  
 m.push\_back(char(rand() % 26 + 97));  
 }  
 if (name) {  
 m.push\_back(' ');  
 }  
}  
  
  
  
  
std:: vector <std:: string> get\_amplua (int n){//создает амплуа  
 std::vector <std::string> amplua (n);  
 std::vector <std:: string> a = {"goalkeeper","midfielder","striker","defender"};  
 for (int i = 0; i < n; ++i) {  
 amplua[i] = a[rand() % 4];  
  
 }  
 return amplua;  
  
}  
  
std:: string get\_club (){//создает клуб  
 int size = rand()%30 + 1;  
 std:: string c;  
 for (int i = 0; i < size; ++i){  
 c.push\_back(char(rand() % 26 + 97));  
 }  
 return c;  
}  
  
  
  
std:: vector <int> get\_Nmatch (int n){//создает кол-во матчей  
 std:: vector <int> a(n);  
 for (int i = 0; i < n; ++i){  
  
 a[i] = rand();  
 }  
 return a;  
}  
  
std:: vector <int> get\_Nball (int n, std::vector <std::string>& ampl){ //создает кол-во мячей  
 std:: vector <int> Nball(n);  
 for (int i = 0; i < n; ++i) {  
 if (ampl[i] == "goalkeeper") {  
 int a = rand();  
 int b = rand();  
 if (a > b){ a = b - a; }  
 else {a = a - b;}  
 Nball [i] = a;  
 }  
 else{  
 Nball [i] = rand();  
 }  
  
 }  
 return Nball;  
}  
  
  
  
class footbal {// описание класса  
 // необходимые переменные

std:: string fio;  
 std:: string club;  
 std:: string amplua;  
 int Nmatch;  
 int Nball;  
public:  
 // методы меняющие значения переменных  
 void put\_fio(std:: string& f){  
 fio = f;  
 }  
 void put\_club(std:: string& c){  
 club = c;  
 }  
 void put\_amplua(std:: string& a ){  
 amplua = a;  
 }  
 void put\_Nmatch(int nm){  
 Nmatch = nm;  
 }  
 void put\_Nball (int nb){  
 Nball = nb;  
 }  
 void print(){//метод для вывода  
 std:: cout << "FIO = "<< fio<< '\n';  
 std:: cout << "club = "<< club << '\n';  
 std:: cout << "amplua = "<< amplua << '\n';  
 std:: cout << "Nmatch = "<< Nmatch << '\n';  
 std:: cout << "Nball = "<< Nball<< '\n';  
  
 }  
  
 bool operator < (footbal b){ // перегрузка оператора меньше, чтобы млжно было сортировать по алфавиту  
 if ((amplua < b.amplua)||((amplua == b.amplua)and(fio < b.fio))){  
 return 1;  
 }  
 return 0;  
 }  
 bool operator > (footbal b){  
 if ((amplua > b.amplua)||((amplua == b.amplua)and(fio > b.fio))){  
 return 1;  
 }  
 return 0;  
 }  
 footbal& operator =(const footbal& b){ // перегрузка оператора присваиваения чтобы оформить функцию swap для класса  
 fio = b.fio;  
 club = b.club;  
 amplua = b.amplua;  
 Nmatch = b.Nmatch;  
 Nball = b.Nball;  
 }  
};  
  
  
  
void quickSort(std::vector <footbal>& array, int left, int right){ // функция быстрой сортировки  
 int i = left;  
 int j = right - 1;  
 footbal m = array[(i + j) / 2];  
 footbal k = array[0];  
 while (i <= j){  
  
 while (array[i] < m) {  
 i++;  
 }  
 while (array[j] > m) {  
 j--;  
 }  
 if (i <= j)  
 {  
 k = array[i];  
 array[i] = array[j];  
 array[j] = k;  
 i++;  
 j--;  
  
 }  
 }  
 if (j > left){  
 quickSort(array, left, j);  
 }  
 if (i < right) {  
 quickSort(array, i, right);  
 }  
  
  
void swap\_f(footbal& a, footbal& b){ // функция для обмена значениями между двумя переменными типа footval  
 footbal tmp;  
 tmp = a;  
 a = b;  
 b = tmp;  
  
}  
  
  
void heapify(std:: vector <footbal>& a, int n, int i) //доп функция для пирамедальной сортировки   
{  
 int largest = i;  
  
 int l = 2\*i + 1; // левый элемент  
 int r = 2\*i + 2; // правый элемент  
  
  
 if (l < n && a[l] > a[largest]) {  
 largest = l;  
 }  
  
 if (r < n && a[r] > a[largest]) {  
 largest = r;  
 }  
 if (largest != i)  
 { swap\_f(a[i],a[largest]);  
  
  
 heapify(a, n, largest);  
 }  
}  
  
void heapSort(std::vector <footbal> a, int n){ //пирамидальная сортировка  
 for (int i = n / 2 - 1; i >= 0; i--) {  
 heapify(a, n, i);  
 }  
  
 for (int i=n-1; i>=0; i--)  
 {  
 swap\_f(a[0], a[i]);  
 heapify(a, i, 0);  
 }  
}  
  
  
  
  
  
int main() {

int a;

std:: cout<< "введите 1 если пирамидальная, введите 0 если быстрая"<<'\n';

std:: cin>> a;  
 size\_t t\_0,t\_1; // переменные для вычисления времени  
 std::vector <size\_t> time\_check; //массив хранящий время исполнения   
 std::vector <int> k; // массив хранящий сколько будет футболистов  
 for (int i = 10; i <= 120010; i+= 2000){  
 k.push\_back(i);  
 }  
for (int j = 0; j < k.size(); ++j) {//Цикл для полного счета при различном количестве футболистлов  
 int n = k[j];  
 std::vector<footbal> f(n);  
 std::vector<std::string> fio = get\_fio(n);  
 std::string club = get\_club();  
 std::vector<std::string> amplua = get\_amplua(n);  
 std::vector<int> Nmatch = get\_Nmatch(n);  
 std::vector<int> Nball = get\_Nball(n, amplua);  
 std::cout<< j <<" "<< k[j]<< '\n';// сделано для удобства, т.к. иначе не понятно на каком этапе реализации находится программа  
 for (int i = 0; i < n; ++i) {  
  
 f[i].put\_fio(fio[i]);  
 f[i].put\_club(club);  
 f[i].put\_amplua(amplua[i]);  
 f[i].put\_Nmatch(Nmatch[i]);  
 f[i].put\_Nball(Nball[i]);  
/\* if (i < 4) {  
 f[i].print();  
 std::cout << '\n';  
 }\*/  
  
 }  
  
 if(a){t\_0 = clock();

heapSort(f, n);

t\_1 = clock();}

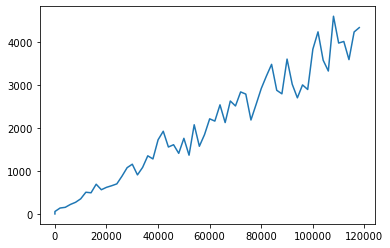
else{

t\_0 = clock;

quickSort(f, 0,f.size());

t\_1 = clock();}  
 time\_check.push\_back(t\_1 - t\_0);  
}  
std:: cout<< '\n';  
for (auto i : time\_check){  
 std:: cout<< i<< ',';  
  
}  
}

Пирамидальная соритровка



x = [0, 10, 2010, 4010, 6010, 8010, 10010, 12010,

14010, 16010, 18010, 20010, 22010, 24010, 26010, 28010,

30010, 32010, 34010, 36010, 38010, 40010, 42010, 44010,

46010, 48010, 50010, 52010, 54010, 56010, 58010, 60010,

62010, 64010, 66010, 68010, 70010, 72010, 74010, 76010,

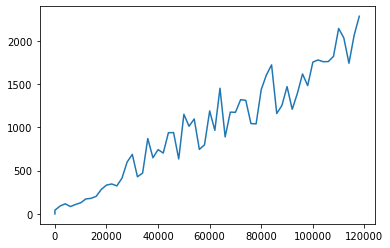
78010, 80010, 82010, 84010, 86010, 88010, 90010, 92010,

94010, 96010, 98010, 100010, 102010, 104010, 106010, 108010,

110010, 112010, 114010, 116010, 118010]

t=[0,56,137,153,220,270,351,504,491,690,562,618,655,698,877,1075,1155,907,1081,1350,1278,1723,1921,1555,1608,1408,1757,1365,2072,1572,1843,2210,2157,2535,2123,2624,2509,2835,2785,2183,2545,2915,3205,3476,2873,2791,3598,3014,2698,2998,2892,3828,4231,3572,3321,4594,3971,4008,3586,4229,4330]

Быстрая сортировка



x= [0 ,10, 2010, 4010, 6010, 8010, 10010, 12010,

14010, 16010, 18010, 20010, 22010, 24010, 26010, 28010,

30010, 32010, 34010, 36010, 38010, 40010, 42010, 44010,

46010, 48010, 50010, 52010, 54010, 56010, 58010, 60010,

62010, 64010, 66010, 68010, 70010, 72010, 74010, 76010,

78010, 80010, 82010, 84010, 86010, 88010, 90010, 92010,

94010, 96010, 98010, 100010, 102010, 104010, 106010, 108010,]

110010, 112010, 114010, 116010, 118010]

t=[0,43,91,117,85,109,129,173,180,203,284,333,346,324,413,600,688,429,472,871,649,743,703,938,940,635,1151,1013,1097,745,799,1190,966,1452,890,1176,1174,1320,1312,1044,1040,1438,1606,1722,1160,1253,1471,1209,1393,1617,1482,1754,1779,1759,1760,1821,2143,2034,1741,2063,2284]