ФГАОУВО «Южно-Уральский государственный университет (НИУ)»

Институт естественных и точных наук

Кафедра «Прикладная математика и программирование»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

по дисциплине «Объектно-ориентированное программирование»

Автор работы

студент группы ЕТ-212

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ M.C.Никитин

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

Работа зачтена с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.К.Демидов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

Челябинск, 2022

# 1 Постановка задачи

I. Реализовать класс

15. Класс упорядоченная последовательность целых чисел (числа могут повторяться).

class SortSeq

{

public:

SortSeq(); // пустая последовательность

~SortSeq();

void add(int x); // добавить число

int size(); // тек. размер

int get(int i); // получить элемент последовательности

void remove(int i); // удалить i-й элемент

void print(); // распечатать состояние объекта

};

II. Реализовать main с тестами для проверки класса (создание объекта и выполнение действий с ним)

# 2 Описание интерфейса класса

class SortSeq{

int n,//текущий размер

nmax;//максимальный размер

double \*mas;//указатель на массив

public:

SortSeq();//пустая последовательность

~SortSeq();//деструктор класса

void add(double x);//добавить число

int size();//текущий размер

double get(int i);//получить элемент последовательности

void remove(int i);//удалить i-ый элемент

void print();//распечатать состояние объекта

};`

# 3 Описание тестов для проверки классов

int main() {

srand(time(0));

SortSeq D;

int count\_numbers = rand() % 5+5;

cout<<format("1.Заполним последовательность {} случаными числами"

"и распечатаем состояние объекта\n", count\_numbers);

for(int i = 0; i < count\_numbers; i ++)

D.add(rand() % 10 + 1);

D.print();

cout<<"2.Текущий размер последовательности:"<<D.size()<<"\n";

int rand\_i4get = rand() % D.size();

cout<<format("3.Получим {} элемент последовательности:", rand\_i4get+1)

<<D.get(rand\_i4get)<<"\n";

int rand\_i4remove = rand() % D.size();

cout<<format("4.Удалим {} эл. послед. и распечатаем состояние объекта\n", rand\_i4remove+1);

D.remove(rand\_i4remove);

D.print();

cout<<"5.Удалим несуществующий элемент последовательности:";

D.remove(D.size()+1);

return 0;

}

**Полученные результаты**

1.Заполним последовательность 7 случайными числами распечатаем состояние объекта

6 8 10 11 13 14 17

2.Текущий размер последовательности:7

3.Получим 6 элемент последовательности:14

4.Удалим 4 эл. послед. и распечатаем состояние объекта

6 8 10 13 14 17

5.Удалим несуществующий элемент последовательности: Неверный индекс

# 4 Листинг реализации класса

SortSeq::SortSeq() {

nmax=8;

n=0;

mas=new double[nmax];

}

SortSeq::~SortSeq() {

delete [] mas;

}

void SortSeq::add(double x) {

if (n == nmax) {

nmax\*= 2;

double \*m = new double[nmax];

for (int i = 0; i<n; ++i)

m[i] = mas[i];

delete [] mas;

mas = m;

}

mas[n++] = x;

for (int i = n-1; i > 0; i--) {

if (mas[i] < mas [i-1]) {

int t = mas[i];

mas[i] = mas[i-1];

mas[i-1] = t;

}

}

}

int SortSeq::size() {

return (n);

}

double SortSeq::get(int i) {

if (i<0 || i>=n) {

cout<<"Неверный индекс\n";

exit(1);

}

else

return (mas[i]);

}

void SortSeq::remove(int i) {

if (i<0 || i>=n) {

cout<<"Неверный индекс\n";

exit(1);

}

else {

for (int ind = i; ind<n-1; ind++)

mas[ind] = mas[ind+1];

n--;

}

}

void SortSeq::print() {

for (int i=0; i < n; i++)

cout<<mas[i]<<" ";

cout<<"\n";

}