Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.2**

Дисциплина: «Основы теории алгоритмов и структуры данных»

Тема: Классы и объекты. Использование конструкторов

Вариант 14

Выполнил:

студент группы РИС-20-2б

Вичугов Алексей Дмитриевич

Проверила:

доцент кафедры ИТАС

Полякова О. А.

Пермь, 2021

**Цель работы**

1. Создание консольного приложения, состоящего из нескольких файлов.
2. Создание объектов с использованием конструкторов.

**Постановка задачи**

1. Определить пользовательский класс.
2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.
3. Определить в классе деструктор.
4. Определить в классе компоненты-функции для просмотра и установки полей (селекторы и модификаторы).
5. Написать демонстрационную программу, в которой продемонстрировать все три случая вызова конструктора: конструктора копирования, конструктора с параметрами и конструктора без параметров.
6. Задание варианта: Пользовательский класс Банкомат, идентификационный номер – int, остаток денег в банкомате – double, максимальная сумма, которую может снять клиент – double.

**Анализ задачи**

1. Определение класса:

class Bank{

int id;

double count;

double max;

public:

int \*s;

Bank();

Bank(int, double, double);

Bank(Bank&);

void Init(int, double, double);

void Show();

~Bank();

int BI();

double BC();

double BM();

};

1. Определение компонентных функций:  
   void Bank::Init(int i, double c, double m){

id = i;

count=c;

max=m;

}

void Bank::Show(){cout << "ID: " << id << endl << "Count: " << count << endl << "Limit: " << max << endl;}

int Bank::BI(){return id;}

double Bank::BC(){return count;}

double Bank::BM(){return max;}

Bank::Bank(){

s=NULL;

id=0;

count=10;

max=5;

}

Bank::Bank(int i, double c, double m){

id=i;

count=c;

max=m;

s=NULL;

}

Bank::Bank(Bank &b){

count=b.BC();

max=b.BM();

id=b.BI();

s=NULL;

}

Bank::~Bank(){

delete s;

}

1. Определение функции make:

void Initer(Bank &b){

int i;

double c, m;

cout << "New data: " << endl;

cout << "ID: "; cin >> i;

cout << "Count: "; cin >> c;

cout << "Limit: "; cin >> m;

b.Init(i, c, m);

cout << endl << "Init B: " << endl;

b.Show();

}

Bank Abonent(){

Bank d;

d.Init(305, 12823.06, 32000);

cout << endl << "D: " << endl;

d.Show();

return d;

}

1. Функция main():

main(){

cout.precision(10);

Bank a;

cout << "A: " << endl;

a.Show();

Bank b(524, 650248.32, 50000.0);

cout << endl << "B: " << endl;

b.Show();

Bank c=b;

cout << endl << "C: " << endl;

c.Show();

Bank d;

d=Abonent();

Initer(b);

Bank e(b);

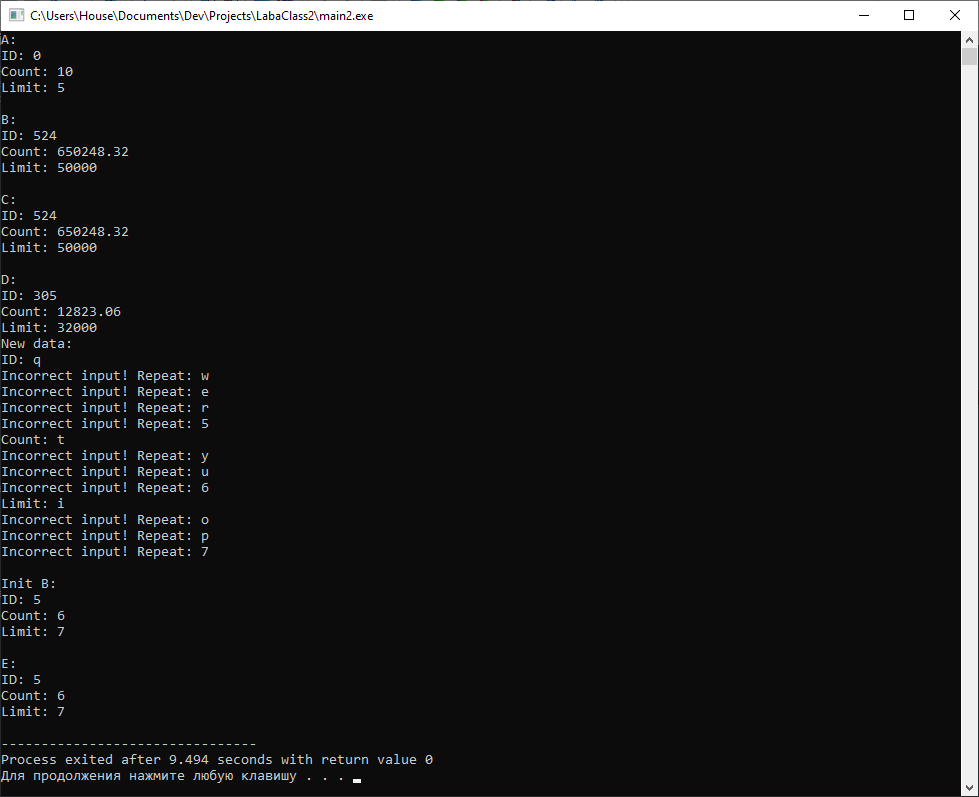
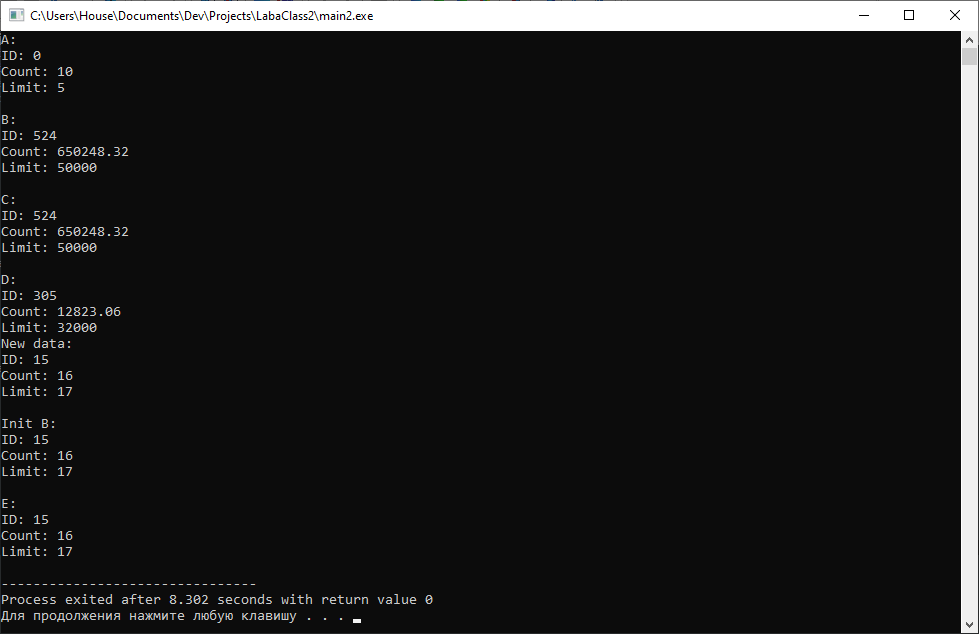
cout << endl << "E: " << endl;

e.Show();

return 0;

}

**Результат работы программы**



**Ответы на вопросы**

1. Конструктор нужен для инициализации полей сразу при создании экземпляра класса.
2. Три – конструктор без параметров, конструктор с параметрами, конструктор копирования.
3. Для освобождения памяти, занимаемой объектом. Деструктор описывается явно, когда в классе имеются объекты, использующие динамическую память.
4. Конструктор без параметров – для инициализации полей объекта класса по умолчанию. Конструктор с параметрами – для инициализации полей заданными значениями. Конструктор копирования – для инициализации полей значениями, равными значениям уже существующего экземпляра класса.
5. Конструктор копирования вызывается при передаче в качестве аргумента экземпляру класса экземпляра этого же класса, а также при передаче объекта в функцию по значению и при возврате объекта из функции.
6. Конструкторы: не возвращают значения, могут быть перегружены, не наследуются, создаются автоматически (без параметров и копирования), могут содержать параметры по умолчанию (только для одного из конструкторов), не могут быть константными, виртуальными или статичными.
7. Деструктор: не имеет аргументов и не возвращает значения, не наследуется, не может быть константным и статическим, но может быть виртуальным.
8. Ко всем.
9. Указатель на этот же объект класса.
10. Методы имеют доступ ко всем полям класса, а внешние функции только к полям со спецификатором доступа public.
11. Конструкторы не возвращают значения.
12. Конструктор без параметров и конструктор копирования.
13. Деструктор не возвращает значения.
14. Конструктор без параметров.
15. Конструктор без параметров.
16. Конструктор с параметрами.
17. Конструктор с параметрами, конструктор копирования.
18. Конструктор с параметрами, конструктор без параметров, конструктор копирования.
19. Конструктор копирования.
20. Вызовом метода set\_name с указанием аргумента.