Министерство образования Республики Беларусь

УО «Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №1**

По дисциплине: “Языки программирования”

Тема: “Классы и объекты в с++”**Вариант №9**

**Выполнил**: студент 2 курса группы ПО-7 Крупенков Михаил Дмитриевич

**Проверила:** Дряпко А. В.

Брест 2021

**Постановка задачи:**

1. Определить пользовательский класс в соответствии с вариантом задания
2. Определить в классе следующие конструкторы: без параметров параметрами, копирования
3. Определить в классе деструктор
4. Определить в классе компоненты-функции для просмотра и установки полей данных
5. Определить указатель на компоненту-функцию
6. Определить указатель на экземпляр класса
7. Написать демонстрационную программу, в которой создаются и разрушаются объекты пользовательского класса и каждый вызов конструктора и деструктора сопровождается выдачей соответствующего сообщения
8. Показать в программе использование указателя на объект и указателя на компоненту-функцию

**Цель:**

Получить практические навыки реализации классов на С++

**Вариант задания:**

Квитанция

Номер – int

Дата – int

Сумма – float

**Код программ:**

**main.cpp**

#include <iostream>  
#include "Ticket.h"  
  
  
int main() {  
 auto\* t1 = new Ticket;  
 *// Использование конструктора без параметра* auto\* t2 = new Ticket(1, 2, 3);  
 *// Использование конструктора с параметрами* auto\* t3 = new Ticket(\*t2);  
 *// Использование конструктора копирования* auto\* t4 = new Ticket(Ticket(3, 3, 7));  
 *// Элизия (умное слово) - копирование анонимного объекта  
 // В таких случаях компилятору разрешается отказаться от вызова  
 // конструктора копирования и просто выполнить стандартный конструктор.* t1->SetNumber(3);  
 std::cout << '\n' << t1->GetNumber();  
 t1->Show();  
  
 t2->SetDate(9);  
 std::cout << '\n' << t2->GetDate();  
 t2->Show();  
  
 t3->SetAmount(t4->GetAmount());  
 std::cout << '\n' << t3->GetAmount();  
  
 *// Указатель на метод* void (Ticket::\*showPtr)() const;  
 showPtr = &Ticket::Show;  
 (t3->\*showPtr)();  
  
 delete t1;  
 delete t2;  
 delete t3;  
 delete t4;  
}

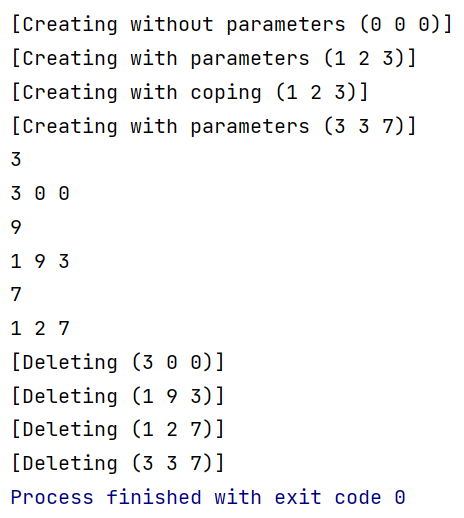
**Ticket.h**

#ifndef **TICKET\_H**#define **TICKET\_H  
  
  
class** Ticket {  
 **int** number;  
 **int** date;  
 **float** amount;  
  
**public**:  
 Ticket();  
 Ticket(**int** number, **int** date, **float** amount);  
 Ticket(**const** Ticket& ticket);  
 ~Ticket();  
  
 **void** Show() **const**;  
  
 **void** SetNumber(**int** newNumber);  
 **int** GetNumber() **const**;  
 **void** SetDate(**int** newDate);  
 **int** GetDate() **const**;  
 **void** SetAmount(**float** newAmount);  
 **float** GetAmount() **const**;  
  
};  
  
#endif *//TICKET\_H*

**Ticket.cpp**

#include **<iostream>**#include **"Ticket.h"**Ticket::Ticket() :  
number(0), date(0), amount(0.0)  
{  
 std::cout << **"\n[Creating without parameters ("** <<  
 number << **' '** << date << **' '** << amount << **")]"**;  
}  
  
Ticket::Ticket(**int** number, **int** date, **float** amount) :  
number(number), date(date), amount(amount)  
{  
 std::cout << **"\n[Creating with parameters ("** <<  
 number << **' '** << date << **' '** << amount << **")]"**;  
}  
  
Ticket::Ticket(**const** Ticket& ticket) :  
number(ticket.number), date(ticket.date), amount(ticket.amount)  
{  
 std::cout << **"\n[Creating with coping ("** <<  
 number << **' '** << date << **' '** << amount << **")]"**;  
}  
  
Ticket::~Ticket() {  
 std::cout << **"\n[Deleting ("** <<  
 number << **' '** << date << **' '** << amount << **")]"**;  
}  
  
  
**void** Ticket::Show() **const** {  
 std::cout << **'\n'** << number << **' '** << date << **' '** << amount;  
}  
  
  
**void** Ticket::SetNumber(**int** newNumber) {  
 number = newNumber;  
}  
  
**int** Ticket::GetNumber() **const** {  
 **return** number;  
}  
  
  
**void** Ticket::SetDate(**int** newDate) {  
 date = newDate;  
}  
  
**int** Ticket::GetDate() **const** {  
 **return** date;  
}  
  
  
**void** Ticket::SetAmount(**float** newAmount) {  
 amount = newAmount;  
}  
  
**float** Ticket::GetAmount() **const** {  
 **return** amount;  
}

**Результат программы:**

****

**Вывод:**

Я получил практические навыки реализации классов на С++