Міністерство освіти та науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет прикладної математики
Кафедра системного програмування і спеціалізованих комп’ютерних систем

ЛАБОРАТОРНА РОБОТА № 4

з дисципліни
“ Програмування-2. Програмування мовою С”
Тема: “КЛАСИ І ОБ’ЄКТИ»

Виконав: Фесенко Д.О.
Студент групи КВ-34
Варіант №20

Київ 2023

Постановка задачі

Написати програму, в якій створюються і знищуються об’єкти, визначені користувачем; виконати виклики конструкторів і деструкторів.

Вказівки до виконання завдання

1. Визначити в програмі класс відповідно до варіанта.
2. Визначити в класі наступні конструктори: без параметрів, з параметрами, копіювання.
3. Визначити в класі деструктор.
4. Визначити в класі методи для перегляду і встановлення полів даних.
5. Визначити вказівник на метод.
6. Визначити вказівник на екземпляр класа.
7. Передбачити розміщення об’єктів як в статичній, так і в динамічній пам’яті, а також створення масивів об’єктів.
8. Написать програму, в якій створюються і знищуються об’єкти користувацього класа і кожний виклик констурктора і деструктора супроводжується видачею відповідного повідомлення (про те, який об’єкт який конструктор або деструктор викликав).
9. Показати в програмі використання вказівника на об’єкт і вказівника на метод.
10. Програма мусить використовувати три файла:
    - заголовочний h-файл з визначенням класа,
    - cpp-файл з реалізацією класа,
    - cpp-файл з демонстраційною програмою.

Завдання за варіантом №20:

Паспорт: (ім’я- char*, вік – int, адреса – char*).

Код програми:

classHeader.h:

```
#ifndef _CLASSREALIZ_CPP_
#define _CLASSREALIZ_CPP_

class Passprt
{
	char* name;
	int age;
	char* address;
public:
	Passprt();//without par
	Passprt(const char* name, int age,const char* address);//with par
	Passprt(const Passprt&);//copy

	char* GetName();
	int GetAge();
	char* GetAddress();

	void SetName(const char*);
	void SetAge(int);
	void SetAddress(const char*);
	void SetAll(const char*, int, const char*);

	void Print();

	~Passprt();

};

#endif // _CLASSREALIZ_CPP_
```

classRealiz.cpp:

```
#include "classHeader.h"
#include <iostream>
#include <cstring>

using namespace std;

#pragma warning(disable : 4996)


Passprt::Passprt() //without par constructor
	{
		name = nullptr;
		address = nullptr;
		age = 0;
		cout << "-->Constructor without parameters is calling.." << this << endl;
	}

Passprt::Passprt(const char* name, int age, const char* address) // with par
constructor
	{
		this->name = new char[strlen(name) + 1];
		strcpy(this->name, name );

		this->age = age;

		this->address = new char[strlen(address) + 1];
		strcpy(this->address, address);
```

```
		cout << "-->Constructor with parameters is calling.." << this << endl;
	}

Passprt::Passprt(const Passprt& other) // copy constructor
	{
		this->name = new char[sizeof(other.name)];
		strcpy(this->name, name);

		this->age = other.age;

		this->address = new char[sizeof(other.address)];
		strcpy(this->address, address);

		cout << "-->Copy constructor is calling.." << this << endl;
	}

char* Passprt::GetName()
	{
		cout << "-->Getter for name is worked!" << endl;
		return name;
	}

int Passprt::GetAge()
	{
		cout << "-->Getter for age is worked!" << endl;
		return age;
	}

char* Passprt::GetAddress()
	{
		cout << "-->Getter for address is worked!" << endl;
		return address;
	}

void Passprt::SetName(const char* name)
	{
		delete[] this->name;
		this->name = new char[strlen(name) + 1];
		strcpy(this->name, name);
		cout << "-->Setter for name is worked!" << endl;
	};

void Passprt::SetAge(int age)
	{
		this->age = age;
		cout << "-->Setter for age is worked!" << endl;
	};

void Passprt::SetAddress(const char* address)
	{
		delete[] this->address;
		this->address = new char[strlen(address) + 1];
		strcpy(this->address, address);
		cout << "-->Setter for address is worked!" << endl;
	};

void Passprt::SetAll(const char* name, int age, const char* address)
	{
		delete[] this->name;
		this->name = new char[strlen(name) + 1];
		strcpy(this->name, name);

		this->age = age;
```

```
            delete[] this->address;
            this->address = new char[strlen(address) + 1];
            strcpy(this->address, address);

            cout << "-->Setter FOR ALL is worked!" << endl;
      };

void Passprt::Print()
      {
            cout << "Passport data:" << endl;
            cout << "name    ~ > " << name << endl;
            cout << "age     ~ > " << age << endl;
            cout << "addres ~ > " << address << endl;
      };

Passprt::~Passprt()
      {
            cout << "-->Destructor is calling.." << this << endl;
            delete[] name;
            delete[] address;
      };
```

demonstrate.cpp:

```
#include <iostream>
#include "classHeader.h"

int main(void)
{
    // Static obj
    Passprt staticPassport("Jane Doe", 28, "456 Elm St.");
    staticPassport.Print();

    // Dynamic obj
    Passprt* dynamicPassport = new Passprt("John Smith", 35, "789 Maple Ave.");
    dynamicPassport->Print();
    delete dynamicPassport;

    // Static array
    Passprt staticPassports[2] = {
        Passprt("Alice", 22, "101 Oak St."),
        Passprt("Bob", 25, "202 Pine St.")
    };
    staticPassports[0].Print();
    staticPassports[1].Print();

    // Dynamic array of objects
    Passprt* dynamicPassports = new Passprt[2];
    dynamicPassports[0].SetAll("Charlie", 27, "303 Cedar St.");
    dynamicPassports[1].SetAll("Dave", 29, "404 Birch St.");
    dynamicPassports[0].Print();
    dynamicPassports[1].Print();
    delete[] dynamicPassports;

    // Obj pointer
    Passprt* passportPtr = new Passprt("John Doe", 30, "123 Main St.");
    passportPtr->Print();
    delete passportPtr;

    // Method pointer
    Passprt passport("Eve", 31, "505 Spruce St.");
    void (Passprt:: * setNamePtr)(const char*) = &Passprt::SetName;
    (passport.*setNamePtr)("Eva Green");
```

```
    passport.Print();

       return 0;
}
```

Тести

```
-->Constructor with parameters is calling..0000003121F9F858
Passport data:
name    ~ > Jane Doe
age     ~ > 28
addres ~ > 456 Elm St.
-->Constructor with parameters is calling..000001FF9F41FFE0
Passport data:
name    ~ > John Smith
age     ~ > 35
addres ~ > 789 Maple Ave.
-->Destructor is calling..000001FF9F41FFE0
-->Constructor with parameters is calling..0000003121F9F8A8
-->Constructor with parameters is calling..0000003121F9F8C0
Passport data:
name    ~ > Alice
age     ~ > 22
addres ~ > 101 Oak St.
Passport data:
name    ~ > Bob
age     ~ > 25
addres ~ > 202 Pine St.
-->Constructor without parameters is calling..000001FF9F415A78
-->Constructor without parameters is calling..000001FF9F415A90
-->Setter FOR ALL is worked!
-->Setter FOR ALL is worked!
Passport data:
name    ~ > Charlie
age     ~ > 27
addres ~ > 303 Cedar St.
Passport data:
name    ~ > Dave
age     ~ > 29
addres ~ > 404 Birch St.
-->Destructor is calling..000001FF9F415A90
-->Destructor is calling..000001FF9F415A78
-->Constructor with parameters is calling..000001FF9F420AC0
Passport data:
```

```
Passport data:
name    ~ > Dave
age     ~ > 29
addres ~ > 404 Birch St.
-->Destructor is calling..000001FF9F415A90
-->Destructor is calling..000001FF9F415A78
-->Constructor with parameters is calling..000001FF9F420AC0
Passport data:
name    ~ > John Doe
age     ~ > 30
addres ~ > 123 Main St.
-->Destructor is calling..000001FF9F420AC0
-->Constructor with parameters is calling..0000003121F9F938
-->Setter for name is worked!
Passport data:
name    ~ > Eva Green
age     ~ > 31
addres ~ > 505 Spruce St.
-->Destructor is calling..0000003121F9F938
-->Destructor is calling..0000003121F9F8C0
-->Destructor is calling..0000003121F9F8A8
-->Destructor is calling..0000003121F9F858
```