

Русенски университет
“Ангел Кънчев”

Задание № 43

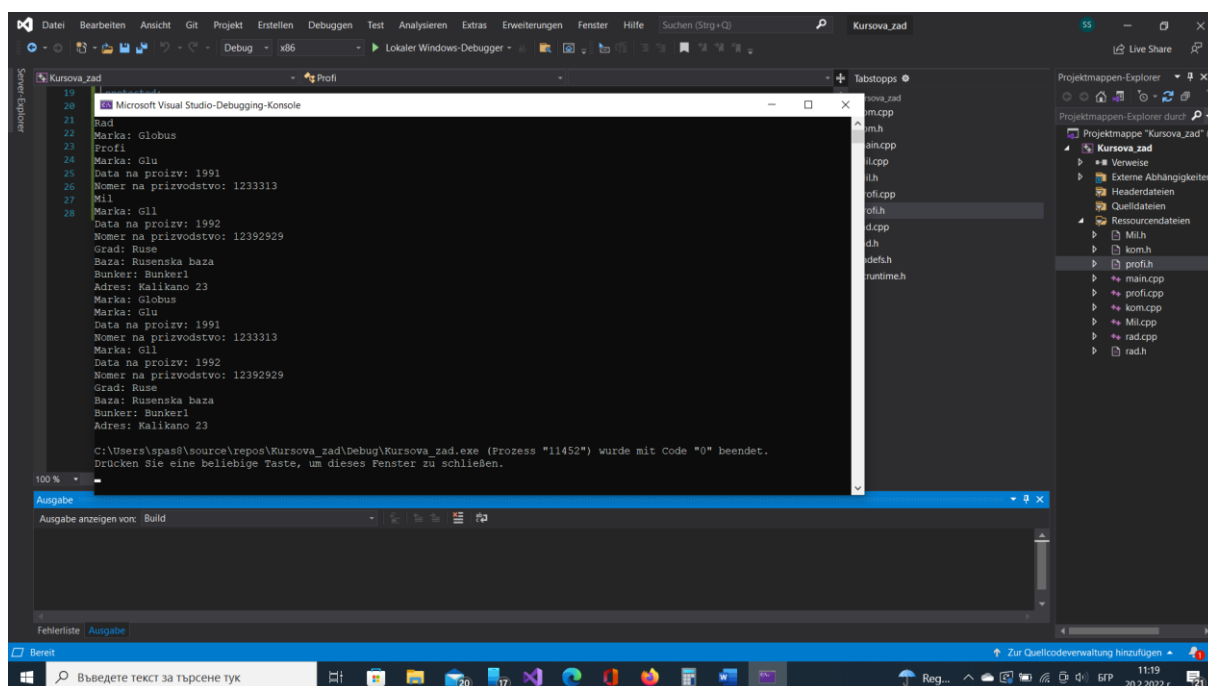
Изготвил : Спас Стефанов Спасов
Студентски № : 213137

Дата :

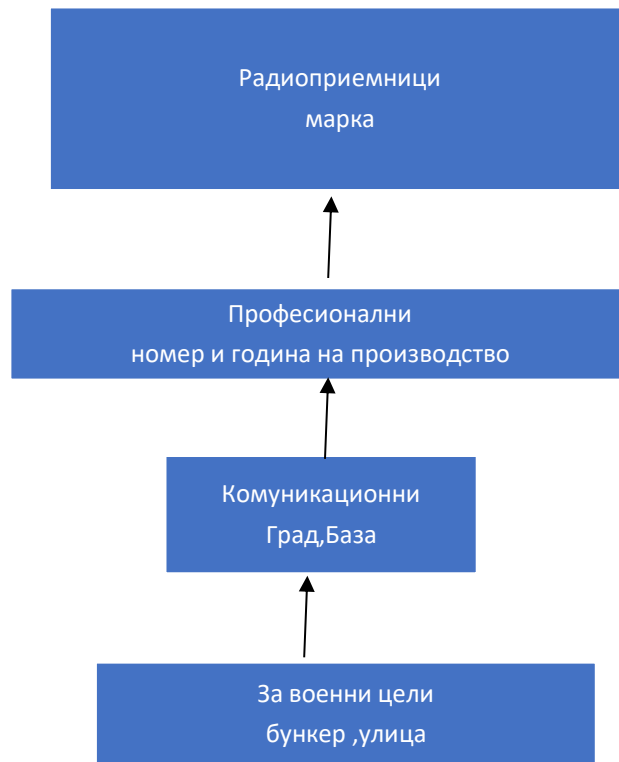
Проверил :

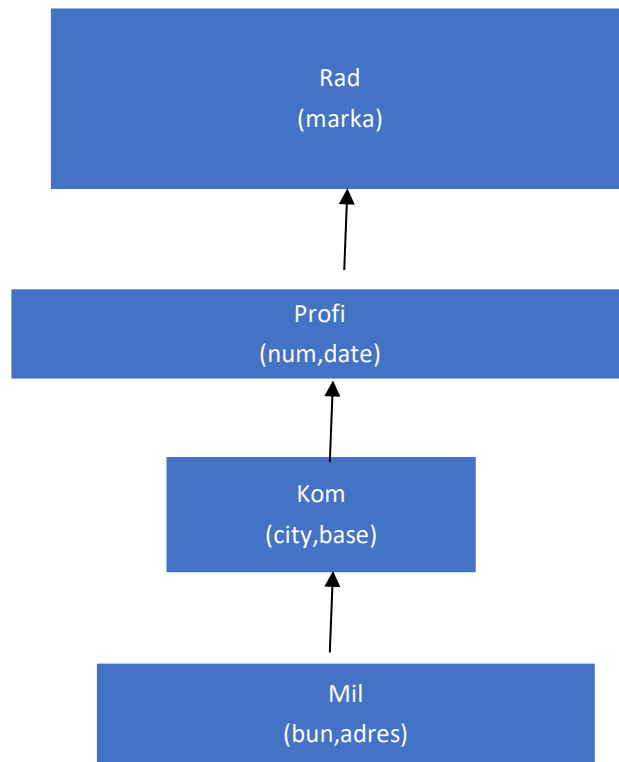
Подпис:

Класификацията да се продължи поне на още две нива. Да се състави йерархия от класове, отразяваща създадената класификация. Да се декларират съответните класове, като всеки клас, с изключение на базовия, да имат поне по 2 собствени атрибута. Да се дефинира виртуална функция, която извежда характеристиките на обект от всеки клас на йерархията. Във функцията main да се изгради едносвързан списък от обекти от произволни класове в йерархията. Да се разработи функция, която обхожда едносвързания списък и извежда информация за признаците на включените в него обекти. Декларациите на всеки клас от йерархията да бъдат оформени в отделни заглавни (.h) файлове. Дефинициите на всеки клас и функцията "main" да бъдат оформени в отделни модули (.cpp файлове). Във всички файлове, съдържащи дефинициите на класовете и функцията "main", чрез директивата #include да се включат съответните заглавни файлове, съдържащи декларациите на класовете. Да се създаде проект, състоящ се от създадените модули. Обяснителната записка (в Word формат) трябва да съдържа заданието, схема на наследяване на класовете (с имена от кода на програмата), пълно описание на декларираните класове от първо и второ ниво (съгласно шаблона по-долу), и код на програмата - всички заглавни файлове (.h) и модули (.cpp). Screen Shot (копие/снимка на екрана) от изпълнението на програмата, доказващ нейната работоспособност. Обяснителната записка трябва да бъде оформена в един файл в Word формат и да бъде изпратена по електронната поща на ръководителя на упражненията. По електронната поща да се изпраща само един файл, съдържащ обяснителната записка.



```
19
20
21
22 Marka: Globus
23 Profi
24 Marka: Glu
25 Data na proizv: 1991
26 Nomer na proizvodstvo: 1233313
27 Mil
28 Marka: Gll
29 Data na proizv: 1992
30 Nomer na proizvodstvo: 12392929
31 Grad: Ruse
32 Baza: Rusenska baza
33 Bunker: Bunker1
34 Adres: Kalikano 23
35 Marka: Globus
36 Marka: Glu
37 Data na proizv: 1991
38 Nomer na proizvodstvo: 1233313
39 Marka: Gll
40 Data na proizv: 1992
41 Nomer na proizvodstvo: 12392929
42 Grad: Ruse
43 Baza: Rusenska baza
44 Bunker: Bunker1
45 Adres: Kalikano 23
46
47 C:\Users\spas8\source\repos\Kursova_zad\Debug\Kursova_zad.exe (Prozess "11452") wurde mit Code "0" beendet.
48 Drücken Sie eine beliebige Taste, um dieses Fenster zu schließen.
```





ИМЕ НА КЛАСА: Rad
НАСЛЕДЯВА ОТ: Profi БАЗОВ ЗА: Ko,Mil

ЧЛЕНОВЕ ДАННИ:
име:marka тип: char*

ИМЕ НА КЛАСА: Profi
НАСЛЕДЯВА ОТ: Kom
ЧЛЕНОВЕ ДАННИ:
име:num тип: int
име:date тип: double

ИМЕ НА КЛАСА:Kom
НАСЛЕДЯВА ОТ: Mil
име:city тип: char*
име:base тип: char*

ИМЕ НА КЛАСА:Kom

Не се наследява.

име:bun тип: char*

име:adres тип: char*

Решение на задачата

```
#ifndef RAD_H
#define RAD_H

class Rad
{
public:
    Rad(void);
    Rad(const char*);
    Rad(Rad&);
    virtual ~Rad(void)
    {
        delete marka;
    }
    char* getmarka(void);
    virtual void answerName(void);
    virtual const char* getClassName();
    void link(Rad*);
    void printlist(Rad*);

protected:
    char* marka;
    Rad* next;
};

#endif // !
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>

#include <cstring>

using namespace std;
#include "rad.h"

Rad::Rad(void)
{
    next = 0;
    marka = new char[8];
    strcpy(marka, "none");
}

Rad::Rad(const char* amarka)
{
    next = 0;
```

```

        marka = new char[strlen(amarka) + 1];
        strcpy(marka, amarka);
    }
Rad::Rad(Rad& arad)
{
    next = 0;
    marka = new char[strlen(arad.marka) + 1];
    strcpy(marka, arad.marka);
}
char* Rad::getmarka(void)
{
    return marka;
}

void Rad::answerName(void)
{
    cout << "Marka: " << marka << endl;
}
const char* Rad::getClassName()
{
    return "Rad";
}

void Rad::link(Rad* pt)
{
    pt->next = next;
    next = pt;
}

void Rad::printlist(Rad* beglist)
{
    for (Rad* np = beglist; np; np = np->next)
        np->answerName();
}

```

```

#ifndef PROFI_H
#define PROFI_H

class Profi :public Rad
{
public:
    Profi(void);
    Profi(const char*, int, double);

    int getnum(void);

    double getdate(void);

    virtual void answerName(void);
    virtual const char* getClassName();

protected:
    int num;
    double date;
}

```

```

};

#endif // !
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>

#include <cstring>

using namespace std;

#include "rad.h"

#include "profi.h"

Profi::Profi(void)
{
    num = 0;
    date = 0;
}
Profi::Profi(const char* amarka, int b, double c) :Rad(amarka)
{
    num = b;
    date = c;
}

int Profi::getnum(void)
{
    return num;
}

double Profi::getdate(void)
{
    return date;
}

void Profi::answerName(void)
{
    Rad::answerName();
    cout << "Data na proizv: " << date << endl;
    cout << "Nomer na prizvodstvo: " << num << endl;
}
const char* Profi::getClassName()
{
    return "Profi";
}

} #ifndef KOM_H
#define KOM_H

class Kom :public Profi

```

```

{
public:
    Kom(const char*, int, double, const char*, const char*);

    ~Kom(void)
    {
        delete city;
        delete base;
    }
    char* getcity(void);

    char* getbase(void);
    virtual void answerName(void);
    virtual const char* getClassName();
protected:
    char* city;
    char* base;
};

```

```

#endif // !KOM_H

```

```

#define _CRT_SECURE_NO_WARNINGS
#include <iostream>

```

```

#include <cstring>

```

```

using namespace std;

```

```

#include "rad.h"

```

```

#include "profi.h"

```

```

#include "kom.h"

```

```

Kom::Kom(const char* amarka, int b, double c, const char* ci, const char* ba) :
Profi(amaraka,b,c)

```

```

{
    city = new char[strlen(ci) + 1];
    strcpy(city, ci);
    base = new char[strlen(ba) + 1];
    strcpy(base, ba);
}

```

```

char* Kom::getcity(void)
{
    return city;
}

```

```

char* Kom::getbase(void)
{
    return base;
}

```



```

void Kom::answerName(void)
{
    Profi::answerName();
    cout << "Grad: " << city << endl;
    cout << "Baza: " << base << endl;
}
const char* Kom::getClassName()
{
    return "Kom";
}
#endif
#define MIL_H

class Mil :public Kom
{
public:
    Mil(const char*, int, double, const char*, const char*, const char*, const
char*);

    ~Mil(void)
    {
        delete bun;
        delete adres;
    }

    char* getbun(void);

    char* getadres(void);
    virtual void answerName(void);
    virtual const char* getClassName();
protected:
    char* bun;
    char* adres;
};

#endif // !MIL_H
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>

#include <cstring>

using namespace std;

#include "rad.h"

#include "profi.h"

#include "kom.h"

#include "Mil.h"

Mil::Mil(const char* amarka, int b, double c, const char* ci, const char* ba, const
char* bu, const char* adr) : Kom(amarka, b, c, ci, ba)
{
    bun = new char[strlen(bu) + 1];

```

```

        strcpy(bun, bu);
        adres = new char[strlen(adr) + 1];
        strcpy(adres, adr);
    }

    char* Mil::getbun(void)
    {
        return bun;
    }
    char* Mil::getadres(void)
    {
        return adres;
    }
    void Mil::answerName(void)
    {
        Kom::answerName();
        cout << "Bunker: " << bun << endl;
        cout << "Adres: " << adres << endl;
    }
    const char* Mil::getClassName()
    {
        return "Mil";
    }
}

```

```

#define _CRT_SECURE_NO_WARNINGS
#include <iostream>

#include <cstring>

using namespace std;

#include "rad.h"

#include "profi.h"
#include "kom.h"
#include "Mil.h"

void main(void)
{
    // Rad* obt[3];
    Rad* p, * pl;
    Rad* d = new Rad("Globus");
    Rad* c = new Profi("Glu", 1233313, 1991);
    Rad* g = new Mil("Gll", 12392929, 1992, "Ruse", "Rusenska
baza", "Bunker1", "Kalikano 23");

    p = d;
    Rad* list = p;
    cout << p->getClassName() << endl;
    p->answerName();
    pl = c;
    p->link(pl);
    p = pl;
}

```

```

cout << p->getClassName() << endl;
p->answerName();
p1 = g;
p->link(p1);
cout << p1->getClassName() << endl;
p1->answerName();

p->printlist(list);
/*obt[0] = g;
obt[1] = d;
obt[2] = c;*/

/*
for (int index = 0; index < 3; index++)
{
    cout << obt[index]->getClassName() << endl;
    obt[index]->answerName();

}*/
delete d;
delete c;
delete g;

//Rad printlist();

```

```

}

```