Универзитет „Св. Кирил и Методиј“ - Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ
И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Аудиториски вежби 5

Динамичка алокација на меморија, композиција и наследување

Напреден развој на софтвер

Содржина

Пример 1/4

protected наследување и protected членови на класата

```
#include <iostream>
#include <cmath>
using namespace std;
float const PI = 3.14;
class Tocka {
private:
    float x, y;
protected:
    float rast(Tocka t) {
        return (sqrt((x - t.x) * (x - t.x) + (y - t.y) * (y - t.y)));
    }
public:
    Tocka(float xx = 0, float yy = 0) {
        x = xx;
        y = yy;
    }
    void set(float xx, float yy) {
        x = xx;
        y = yy;
    }
    friend ostream & operator<<(ostream & out, Tocka &t) {
        return out << "(" << t.x << "," << t.y << ")";
    }
};
```

Пример 2/4

protected наследување и protected членови на класата

```
class Krug: public Tocka {
protected:
    float r;
public:
    Krug(float xx = 0, float yy = 0, float rr = 1) :
        Tocka(xx, yy) {
        r = rr;
    }
    bool leziVo(Tocka t) {
        return (rast(t) < r);
    }
    float plostina() {
        return r * r * PI;
    }
};
```

Пример 3/4

protected наследување и protected членови на класата

```
class Sfera: protected Krug {
private:
    float z;
public:
    Sfera(float xx, float yy, float zz, float rr) :
        Krug(xx, yy, rr) {
        z = zz;
    }
    void set(float xx, float yy, float zz) {
        Tocka::set(xx, yy);
        z = zz;
    }
    float plostina() {
        return 4. * Krug::plostina();
    }
    float volumen() {
        return 4. / 3 * Krug::plostina() * r;
    }
};
```

Пример 4/4

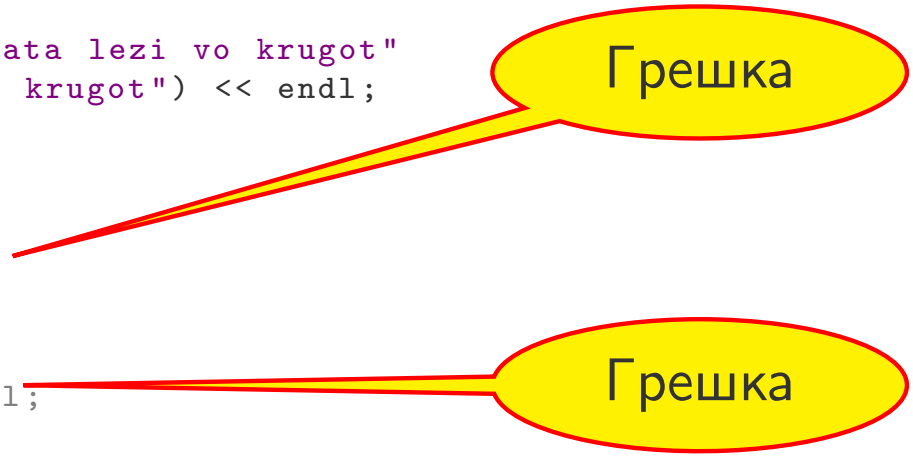
protected наследување и protected членови на класата

```
int main() {
    Tocka t1(5, 5);
    Krug k(5, 5, 3);
    Sfera s(5, 5, 5, 3);

    cout << k << endl; // se povikuva operatorot << nasleden od tocka
    cout << t1 << endl;
    k.set(7, 7);

    cout << (k.leziVo(t1) ? "Tockata lezi vo krugot"
            : "Tockata ne lezi vo krugot") << endl;
    cout << k.plostina() << endl;
    t1 = k;
    cout << t1 << endl;

    //cout << k.rast(t1) << endl;
    cout << s.plostina() << endl;
    cout << s.volumen() << endl;
    //cout << s.leziVo(t1) << endl;
    return 0;
    return 0;
}
```



Задача 1

Да се напише класа за работа со функција `f(x, y)` претставена како множество од точки во 3Д простор (динамички алоцирана меморија за објекти од класата `Tocka3D` која исто така треба да се имплементира).
Функцијата f треба да ги поддржува оперторите += за додавање нова точка во множеството точки, « за печатење на функцијата `f` и `[i]` за промена на точката на позиција `i`. Исто така, класата треба да има метод кој ќе врши интерполација на точките од функцијата и ќе ја враќа новодобиената интерполирана функција.

Задача 1
Решение 1/5

```
#include <iostream>
#include <cstdlib>
#include <cmath>
using namespace std;
class Tocka {
private:
    double x, y;
public:
    Tocka(double xx = 0, double yy = 0) {
        x = xx;
        y = yy;
    }
    virtual void set(double xx, double yy) {
        x = xx;
        y = yy;
    }
    double getx() const { return x; }
    double gety() const { return y; }
    Tocka interpolate(const Tocka &t) const {
        Tocka i;
        i.x = (x + t.x) / 2;
        i.y = (y + t.y) / 2;
        return i;
    }
    friend ostream & operator<<(ostream &out, const Tocka &t) {
        return out << '(' << t.x << ',' << t.y << ')';
    }
};
```

Задача 1
Решение 2/5

```
class Tocka3D: public Tocka {
private:
    double z;
public:
    Tocka3D(double xx = 0, double yy = 0, double zz = 0) :
        Tocka(xx, yy) {
        z = zz;
    }
    void set(double xx, double yy, double zz) {
        Tocka::set(xx, yy);
        z = zz;
    }
    double getz() { return z; }

    Tocka3D interpolate(const Tocka3D &t) const {
        Tocka3D i;
        i.set((getx() + t.getx()) / 2, (gety() + t.gety()) / 2, (z + t.z) / 2);
        return i;
    }

    friend ostream & operator<<(ostream &out, const Tocka3D &t) {
        return out << '(' << t.getx() << ',' << t.gety() << ',' << t.z << ')';
```

Задача 1
Решение 3/5

```
    }
};
class F {
private:
    Tocka3D *tocki;
    int max;
    int broj;
public:
    F(int m = 0) {
        max = m;
        tocki = new Tocka3D[m];
        broj = 0;
    }
    F(const F &f) {
        max = f.max;
        broj = f.broj;
        tocki = new Tocka3D[max];
        for (int i = 0; i < broj; i++)
            tocki[i] = f.tocki[i];
    }
    F operator=(const F &f) {
        if (this != &f) {
            max = f.max;
            broj = f.broj;
            delete[] tocki;
            tocki = new Tocka3D[max];
            for (int i = 0; i < broj; i++)
                tocki[i] = f.tocki[i];
        }
```

Задача 1
Решение 4/5

```
    return *this;
}
F operator +=(const Tocka3D &t) {
    if (broj < max) {
        tocki[broj] = t;
        broj++;
    } else
        cout << "nema mesto za nova tocka\n";
    return *this;
}
Tocka3D & operator [](int i) {
    if (i >= 0 && i < broj)
        return tocki[i];
    else {
        cout << "nevaliden indeks\n";
        exit(1);
    }
}
F interpolacija() {
    F f(max * 2 - 1);
    f += tocki[0];
    for (int i = 0; i < broj - 1; i++) {
        f += tocki[i].interpolate(tocki[i + 1]);
        f += tocki[i + 1];
    }
    return f;
}
~F() {
    delete[] tocki;
}
friend ostream& operator<<(ostream &out, const F &f) {
    for (int i = 0; i < f.broj - 1; i++)
        out << f.tocki[i] << "->";
    return out << f.tocki[f.broj - 1];
```

Задача 1
Решение 5/5

```
    }
};
int main() {
    Tocka3D t(1, 1, 1);
    F f(5);
    for (int i = 0; i < 3; i++) {
        f += t;
        t.set(2 * (i + 1), 2 * (i + 1), 2 * (i + 1));
    }
    cout << f << endl;
    cout << t << endl;
    F fi;
    fi = f.interpolacija();
    cout << fi << endl;
    fi[0] = t;
    fi += t;

    cout << fi << endl;
    cout << fi[0] << ' ';
    cout << fi[7] << endl;
    return 0;
}
```

Задача 2

Да се развие класа за работа со производи со можност за менување на цената и пресметка на заштедата при можен избор од два производи. Да се преоптовари операторот < за споредба на продажната цена на два производи. Од оваа класа да се изведе класа за работа со производи на процентуален попуст за кои цената која се наплаќа при купување се намалува за соодветниот попуст. Да се напише функција која на влез ќе прима два производи (од ист или различен вид) и ќе го советува купувачот кој е поефтин и колкава е заштедата.

Задача 2
Решение 1/4

```
#include <iostream>
#include <stdlib.h>
#include <cmath>
using namespace std;

class Sale {
private:
    double price;
public:
    Sale();
    Sale(double thePrice);
    double getPrice() const;
    void setPrice(double newPrice);
    virtual double bill() const;
    double savings(const Sale& other) const;
    friend bool operator <(const Sale& first, const Sale& second);
};
Sale::Sale() :
    price(0) {
}
Sale::Sale(double thePrice) {
    if (thePrice >= 0)
        price = thePrice;
    else {
        cout << "Error: Cannot have a negative price!\n";
        exit(1);
    }
}
```

Задача 2
Решение 2/4

```
double Sale::bill() const {
    return price;
}
double Sale::getPrice() const {
    return price;
}
void Sale::setPrice(double newPrice) {
    if (newPrice >= 0)
        price = newPrice;
    else {
        cout << "Error: Cannot have a negative price!\n";
        exit(1);
    }
}
double Sale::savings(const Sale& other) const {
    return (bill() - other.bill());
}
bool operator <(const Sale& first, const Sale& second) {
    return (first.bill() < second.bill());
}
```

Задача 2
Решение 3/4

```
class DiscountSale: public Sale {
private:
    double discount;
public:
    DiscountSale();
    DiscountSale(double thePrice, double theDiscount);
    //Discount is expressed as a percent of the price.
    //A negative discount is a price increase.
    double getDiscount() const {
        return discount;
    }
    void setDiscount(double newDiscount) {
        discount = newDiscount;
    }
    double bill() const {
        double fraction = discount / 100;
        return (1 - fraction) * getPrice();
    }
};
DiscountSale::DiscountSale() :
    Sale(), discount(0) {
}
DiscountSale::DiscountSale(double thePrice, double theDiscount) :
    Sale(thePrice), discount(theDiscount) {
}
```

Задача 2
Решение 4/4

```
void compare(const Sale &s1, const Sale &s2) {
    if (s1 < s2) {
        cout << "Prviot e poeftin.\n";
    } else
        cout << "Vtoriot e poeftin.\n";
    cout << "Zastedata e " << fabs(s2.savings(s1)) << " den.\n";
}

int main() {
    //One item at 10.00.
    Sale simple(10.00);
    //One item at 11.00 with a 10% discount.
    DiscountSale discount(11.00, 10);
    compare(simple, discount);
    return 0;
}
```

Задача 3

Да се состави хиерархија на класи за конверзии од еден систем на единици во друг. Основната класа чува две променливи, `val1` и `val2`, кои се однесуваат на иницијалната и конвертираната вредност. Да се имплементираат функции `getinit()` and `getconv()`, кои ја враќаат иницијалната односно конвертираната вредност соодветно. Функцијата која ја прави самата конверзија, `compute()`, е чиста виртуелна функција. Да се додадат две изведени класи за конверзија на литри во галони и фаренхајти во целзиусови степени.

Задача 3
Решение 1/3

```
#include <iostream>
using namespace std;

class convert {
protected:
    double val1; // initial value
    double val2; // converted value
public:
    convert(double i) {
        val1 = i;
    }
    double getconv() {
        return val2;
    }
    double getinit() {
        return val1;
    }
    virtual void compute() = 0;
};
```

Задача 3
Решение 2/3

```
// Liters to gallons.
class l_to_g: public convert {
public:
    l_to_g(double i) :
        convert(i) {
    }
    void compute() {
        val2 = val1 / 3.7854;
    }
};
// Fahrenheit to Celsius
class f_to_c: public convert {
public:
    f_to_c(double i) :
        convert(i) {
    }
    void compute() {
        val2 = (val1 - 32) / 1.8;
    }
};
// Feet to meters
class f_to_m: public convert {
public:
    f_to_m(double i) :
        convert(i) {
    }
    void compute() {
        val2 = val1 / 3.28;
    }
};
```

Задача 3
Решение 3/3

```
int main() {
    convert *p; // pointer to base class
    l_to_g lgob(4);
    f_to_c fcob(70);
    p = &lgob;
    cout << p->getinit() << " liters is ";
    p->compute();
    cout << p->getconv() << " gallons\n"; // l_to_g
    p = &fcob;
    cout << p->getinit() << " in Fahrenheit is ";
    p->compute();
    cout << p->getconv() << " Celsius\n"; // f_to_c
    return 0;
}
```

Задача 4

Да се напише класа `Telefon` со која ќе се овозможи работа со податоци за телефони. За секој телефон се чуваат година на производство (int), почетна цена (int) и модел на производот (низа од 40 знаци).
Од класата `Telefon` да се изведе класа `Mobilen` за кој дополнително се чуваат информации за ширината (float) и висината (float), дебелина (float) во центиметри.
Од класата `Telefon` да се изведе и класа `Fiksen` за кој дополнително ќе се чува информација за неговата тежина во грамови (int). За секоја од класите да се напише соодветен конструктор, set и get методи и да се преоптовари операторот « за печатење.
За секоја од изведените класи да се обезбеди функција `presmetajVrednost()` за пресметка на моменталната цена како:

- За мобилен телефон: 0.95% од цената ако е произведен пред 2010 година и цената зголемена за 5% ако е потенок од 0.7cm
- За фиксен телефон: 0.98% од цената ако е произведен пред 2009 година и цената зголемена за 7% ако е полесен од 100гр

Да се напише функција која на влез прима низа од покажувачи кон класата `Telefon` и ја печати цената на телефонот со најмала цена. Да се напише `main` функција во која ќе се тестираат имплементираните функции во класите.

Задача 4

Решение 1/4

```
#include<iostream>
#include<cstring>
using namespace std;

class Telefon {
protected:
    int god_proizvodstvo;
    int pocetna_cena;
    char proizvoditelModel[40];
public:
    Telefon() {
    }
    Telefon(int g, int p, const char *m) {
        god_proizvodstvo = g;
        pocetna_cena = p;
        strncpy(proizvoditelModel, m, 39);
        proizvoditelModel[39] = '\0';

    }
    virtual float presmetajVrednost()= 0;
    friend bool operator<(Telefon &t1, Telefon &t2) {
        return t1.presmetajVrednost() < t2.presmetajVrednost();
    }
    void Pecati() {
        cout << "God:" << god_proizvodstvo << endl;
        cout << "Poc:" << pocetna_cena << endl;
        cout << "Model:" << proizvoditelModel << endl;
    }
};
```

Задача 4
Решение 2/4

```
class Mobilen: public Telefon {
private:
    float visina;
    float sirina;
    float debelina;

public:
    Mobilen() {
    }
    Mobilen(int g, int p, const char *m, float v, float s, float d) :
        Telefon(g, p, m), visina(v), sirina(s), debelina(d) {
    }
    float presmetajVrednost() {
        double cena = pocetna_cena;
        if (god_proizvodstvo < 2010)
            cena *= 0.95;
        if (debelina < 0.7)
            cena *= 1.05;
        return cena;
    }
};
```

Задача 4
Решение 3/4

```
class Fiksen: public Telefon {
private:
    int masa;
public:
    Fiksen() {
    }
    Fiksen(int g, int p, const char *m, int t) :
        Telefon(g, p, m) {
        masa = t;
    }
    float presmetajVrednost() {
        double cena = pocetna_cena;
        if (god_proizvodstvo < 2009)
            cena *= 0.95;
        if (masa < 100)
            cena *= 1.07;
        return cena;
    }
};
```

Задача 4

Решение 4/4

```
void minCena(Telefon **t, int br) {
    Telefon* temp = t[0];
    for (int i = 1; i < br; i++) {
        if ((*t[i]) < *temp)
            temp = t[i];

    }
    cout << temp->presmetajVrednost() << endl;

}
int main() {
    Mobilen m1(2005, 1500, "mod1", 10, 5, 1.5);
    Mobilen m2(2011, 1500, "mod2", 7, 5, 0.5);

    Fiksen f1(2009, 10000, "mod3", 80);
    cout << "M1:" << m1.presmetajVrednost() << endl;
    cout << "M2:" << m2.presmetajVrednost() << endl;
    cout << "F1:" << f1.presmetajVrednost() << endl;

    Telefon** telefoni;
    telefoni = new Telefon*[3];
    telefoni[0] = &m1;
    telefoni[1] = &m2;
    telefoni[2] = &f1;
    minCena(telefoni, 3);
    return 0;
}
```

Задача 5

Компанијата ФИНКИ-Фото нуди фотоапарати од различни производители и со различни карактеристики. За потребите на компанијата треба да се развијат класи кои ги опишуваат фотоапаратите кои таа ги нуди. За таа цел, треба да се развие класа `Fotoaparat`, во која се чуваат информации за:

- модел (низа од 30 знаци),
- основна цена (реален број),
- резолуција изразена во мегапиксели (цел број).

ФИНКИ-Фото располага со два вида фотоапарати за кои треба да се изведат две посебни класи `DSLR` фотоапарати и `Kompaktni` фотоапарати. За `DSLR` фотоапаратите се чува дополнителна информација за видот на објективот (низа од 20 знаци) и цена на објективот (реален број), а за компактните фотоапарати дополнително се чува информација за видот на зумот (boolean променлива – вредност true ако станува збор за оптички зум, вредност false – за дигитален зум).
За секоја од дефинираните класи да се напише конструктор кој ќе ги иницијализира сите атрибути за апаратите и да се преоптовари операторот << за печатење.

Задача 5

За секоја од класите `DSLR` и `Kompaktni` треба да се имплементира метод за пресметување на цената на фотоапаратите според следниот критериум:

- За DSLR фотоапарати:
  - Основната цена се зголемува за 15% ако имаат резолуција не помала од 15 мегапиксели
  - потоа се додава цената на објективот.
- За компактните фотоапарати:
  - Основната цена се зголемува за 12% ако имаат резолуција не помала од 10 мегапиксели
  - Потоа се пресметува покачување на цената уште за 10% ако станува збор за оптички зум

Да се преоптовари операторот $<$ за споредба на фотоапарати, кои може да бидат од различен вид, според нивната цена. Да се напише и фунција (`najmalaCena`) која како аргумент прима низа од покажувачи кон разни видови фотоапарати, како и нивниот број, а ја печати цената на фотоапаратот со најмала цена.

Задача 5
Решение 1/3

```
class Fotoaparat {
private:
    char model[30];
    double osnCena;
    int rezolucija;
public:
    Fotoaparat(char const *m = "", double o = 0.0, int r = 0) {
        strncpy(model, m, 29);
        model[29] = '\0';
        osnCena = o;
        rezolucija = r;
    }
    int getRezolucija() const {
        return rezolucija;
    }
    int getOsnCena() const {
        return osnCena;
    }
    virtual double cena()= 0;
    friend bool operator<(Fotoaparat &f1, Fotoaparat &f2) {
        return f1.cena() < f2.cena();
    }
    friend ostream& operator<<(ostream& o, const Fotoaparat& f) {
        o << "Model:" << f.model << endl;
        o << "Osnovna cena:" << f.osnCena << endl;
        o << "Rezolucija:" << f.rezolucija << endl;
        return o;
    }
};
```

Задача 5
Решение 2/3

```
class DSLR: public Fotoaparat {
private:
    char vidObjektiv[20];
    double cenaObjektiv;

public:
    DSLR(char const *m = "", double o = 0.0, int r = 0, char const *vo = "",
            double co = 0.0) :
        Fotoaparat(m, o, r), cenaObjektiv(co) {
        strncpy(vidObjektiv, vo, 19);
        vidObjektiv[19] = '\0';
    }
    friend ostream& operator<<(ostream& o, const DSLR& f) {
        o << (Fotoaparat&) f;
        o << "Vid na objektivot:" << f.vidObjektiv << endl;
        o << "Cena na objektivot:" << f.cenaObjektiv << endl;
        return o;
    }
    double cena() {
        if (getRezolucija() >= 15)
            return getOsnCena() * 1.15 + cenaObjektiv;
        return getOsnCena() + cenaObjektiv;
    }
};
```

Задача 5
Решение 3/3

```
class Kompakten: public Fotoaparat {
private:
    bool zum;
public:
    Kompakten(char const *m = "", double o = 0.0, int r = 0, bool z = true) :
        Fotoaparat(m, o, r), zum(z) {
    }
    friend ostream& operator<<(ostream& o, const Kompakten& k) {
        o << (Fotoaparat&) k;
        o << "Zum: ";
        if (k.zum)
            o << "opticki" << endl;
        else
            o << "digitalen" << endl;
        return o;
    }
    double cena() {
        double tmp = getOsnCena();
        if (getRezolucija() >= 10)
            tmp *= 1.12;
        if (zum)
            tmp *= 1.1;
        return tmp;
    }
};
```

Материјали и прашања

Предавања, аудиториски вежби, соопштенија
courses.finki.ukim.mk

Изворен код на сите примери и задачи
bitbucket.org/tdelev/finki-nrs

Прашања и одговори
qa.finki.ukim.mk