Programmeerimine keeles C++

Praktikum 4: viidad, viited ja nende kasutamine

Üldised nõuded

Palun teha teek `libneljas.a`, mille sisu on kätte saadav päise `neljas.h` kaudu. Sellesse teeki lisage kõik ülesandes nõutud lahendused. Soovitatav kasutada eri ülesannete jaoks erinevaid faile. Kindlasti dokumenteerige tulemus ja lisage töötav `Makefile`. Jällegi palun kasutage etteantud meetodite ja liikmete nimesid.

Ülesanne 1 – Kahe muutuja väärtuse vahetamine viite ja viidaga

Kirjutage funktsioonid:

```
template<class T> void swap_ref (T &a, T &b);

template<class T> void swap_ptr (T *a, T *b);
```

Nende funktsioonide eesmärk on vahetada ära sisendina antud väärtused.

Näiteks nii:

```
int a = 5, b = 7;
swap_ref<int> (a, b);         // nüüd a == 7 ja b == 5
swap_ptr<int> (&a, &b);       // nüüd a == 5 ja b == 7
```

Lahenduse eest saab 2 punkti.

Ülesanne 2 – Funktsioon ruutvõrrandi lahendamiseks

Kirjutage funktsioon:

```
bool solve (double a, double b, double c, double &x1, double &x2);
```

Parameetritega `a, b` ja `c` on määratud ruutvõrrand $ax^2+bx+c=0$. Funktsioon tagastab väära tõeväärtuse, kui antud võrrand ei lahendu. Vastasel juhul tagastab tõese väärtuse ning kirjutab parameetritesse `x1` ja `x2` lahendid. Lahendite puudumisel `x1` ja `x2` väärtust ei muudeta.

Pisut näidiskoodi:

```
double a = 1.0, b = -2.0, c = 1.0;
double solution1, solution2;              // lahendused võiksid olla 1 ja 1
if (solve (a, b, c, solution1, solution2))
      cout << "Solutions are: " << solution1 << ", " << solution2 << endl;
else
      cout << "There are no solutions in real numbers!“ << endl;
```

Lahenduse eest saab 1 punkti.

Ülesanne 3 – Viidatud tippudega sirglõik

Kirjutage klassimall `template<class T> DynamicLine`, mis esitab sirglõiku üle antud vektoritüübi, kusjuures tipud on määratud viitadega. Järgneb klassi kirjeldus.

Klassi liige	**Kirjeldus**
`T *p1;`	Viit vektori esimesele tipule (avalikult nähtav)
`T *p2;`	Viit vektori teisele tipule (avalikult nähtav)
`DynamicLine ()`	Vaikekonstruktor, väärtustab viidad nullidega
`DynamicLine (T *v1, T *v2)`	Parameetritega konstruktor
`DynamicLine (const DynamicLine& l)`	Koopiakonstruktor
`operator= (const DynamicLine& l)`	Omistamisoperaator
`~DynamicLine ()`	Destruktor

Idee klassi taga on järgmine. Kujutame ette, et me kirjutame joonistusprogrammi, kus maha märgitud tippe saab hiirega ringi vedada. Me soovime, et sirglõigud, mille otstippe liigutatakse, uueneksid automaatselt. Selleks kirjutame sirglõigu klassi, mis ei sisalda enda koopiaid koordinaatidest, vaid viitavad olemasolevatele tippudele.

Defineerige koopiakonstruktor ja omistamisoperaator, mis koopia loomisel jätavad otstipu viidad samaks (ei loo uusi tippe samade koordinaatidega). Konstruktoris ei võeta tippude jaoks mälu ning destruktoris ei tohi mingil juhul tippe kustutada, sest teised sirglõigud võivad neid veel kasutada.

Tööd klassiga illustreerib järgmine koodilõik:

```
Vector2 p1 (1.0, 2.0); Vector2 p2 (0,0); Vector2 p3 (5.0, 2.0);
DynamicLine<Vector2> l1 (&p1, &p2); // lõik p1 ja p2 vahel
DynamicLine<Vector2> l2 = l1;       // kopeerime lõigu l1 (tipud jäävad samaks)
l2.p2 = &p3;                        // määrame l2 teise otstipu
p1.x = 2.0;                         // liiguvad sirgete l1 ja l2 otstipud
```

Lahendamist aitab järgmiste materjalidega tutvumine:

http://www.learncpp.com/cpp-tutorial/911-the-copy-constructor-and-overloading-the-assignment-operator/

http://www.learncpp.com/cpp-tutorial/912-shallow-vs-deep-copying/

Lahenduse eest saab 3 punkti.

Ülesanne 4 – Lihtne dünaamiline massiiv

Kirjutage klassimall template `<class T> MyArray`, mis esitab dünaamilise suurusega massiivi. Massiivi saab elemente lisada. Lisamine on lubatud realiseerida ebaefektiivselt – iga uue elemendi saabumisel haaratakse uus tükk mälu, kuhu andmed ära mahuvad, andmed kopeeritakse sinna ning vana koopia kustutatakse. Selle klassi puhul peavad koopiakonstruktor ja omistamisoperaator andmetest koopia tegema (et koopia ei jääks viitama samadele andmetele). Samuti tuleb

destruktoris andmed kustutada. Pange tähele, et see klass vastutab oma viitade eest täielikult ise, samas kui eelmise ülesande sirglõik vaid salvestas talle etteantud tippe.

Realiseerige selline klass:

Klassi liige	**Kirjeldus**
`T *content;`	Massiivi sisu (viit andmetele, avalikult nähtav)
`unsigned int size;`	Massiivi elementide arv (ei ole avalikult nähtav)
`MyArray ();`	Vaikekonstruktor, väärtustab viida ja arvu nullidega
`MyArray (const MyArray& a)`	Koopiakonstruktor
`operator= (const MyArray& a)`	Omistamisoperaator
`~MyArray ()`	Destruktor
`unsigned int getSize ();`	Tagastab elementide arvu massiivis
`void addElement (T element);`	Lisab antud elemendi massiivi lõppu
`T& operator[] (unsigned int i);`	Tagastab massiivi i-nda elemendi

Näidisprogramm:

```
MyArray<int> numbers;               // jada arvudest
numbers.addElement (5);             // lisame ühe elemendi
numbers.addElement (11);            // ja veel ühe
MyArray<int> copy = numbers;        // kopeerime jada
copy.addElement (13);               // lisame koopiasse elemendi
if (numbers.getSize () != 2 || copy.getSize () != 3) // kontrollime suuruseid
      cout << "Copy failed - wrong sizes!" << endl;
if (numbers[0] != copy[0] || numbers[1] != copy[1]) // kontrollime sisu
      cout << "Copy failed - wrong elements!" << endl;
```

Mäluhaldust aitab teha see materjal: http://www.cplusplus.com/doc/tutorial/dynamic.html

Lahendus annab 4 punkti.

Lisaülesanne 1 – Kui tipud lähevad kaduma

Lisaülesande lahendusest ma seekord programmi ei oota (samas võite loomulikult proovida oma ideid realiseerida). Selle ülesande lahendus kirjutage tekstifaili lisayl.txt. Pange see fail oma lahendusega kaasa, soovitatavalt juurkausta (Makefile kõrvale). Vastake järgmistele küsimustele, arvestades kontekstina 3. ülesande lahendust.

1) Mis juhtub, kui mõni tipp, millele sirglõigud viitavad, hävineb?
2) Kuidas võiks tipu destruktor teada, millistes sirglõikudes ta esineb?
3) Mida võiks tipu destruktor teha, et olukorda parandada?

Lahendust hindan selle järgi, kui veenvalt see kirja on pandud ning kui hästi see lahendus töötada võiks. Loov lähenemine on igati tervitatav.

Lahenduse eest saab kuni 2 lisapunkti.

Üldised tingimused

Tähtis! Loe läbi ülesannete vormistamise tingimused aine veebilehelt! Küsimustega pöörduda aine listi või praktikumijuhendaja poole. Tähtaeg on praktikumi toimumisnädala pühapäeva õhtu kell 23:59 (aega on 7 päeva).