Laborator 3 – Noţiuni introductive despre clase

Obiective:

- Definirea de clase, obiecte

- Explicarea noțiunilor de constructor, constructor de copiere, destructor, autoreferință

Exerciții

I. Scrieți un program care definește un tip de date abstract Stivă, implementează operațiile care se pot realiza asupra lui și exemplifică folosirea acestora. Stiva va conține elemente de tip întreg.
   a) Definiți tipului de date abstract Stivă
   b) Implementați operațiile atașate tipului Stivă
   c) Creați obiecte de tip Stivă și exemplificați folosirea metodelor implementate
   d) Ce se întâmplă dacă se rulează următorul cod?

```
int main() {
   Stiva s; //declaratie unui obiect de tip stiva
   s.push(7);

   Stiva sc = s;
   s.display();
   sc.display();

   sc.pop();
   s.display();
   sc.display();
   return 0;
}
```

Indicații:

a. Pentru definirea tipului de date abstract de date Stivă se va defini clasa Stivă care va fi implementă folosind tablouri, adică atributele stivei vor fi numărul maxim de elemente pe care îl poate conține stiva, tabloul cu elemente de tip întreg și un număr care stochează numărul de elemente curente din stivă. De asemenea se vor specifica prototipurile metodelor membre clasei (comportamentul clasei). La metodele care caracterizează clasa (isEmpty(), isFull(), push(), pop(), display()) se vor atașa două metode speciale care permit crearea obiectelor de tip Stivă – constructor – și dealocarea spațiului de memorie ocupat de un obiect de tip Stiva – destructor –. Descrierea (definirea) clasei Stivă se va realiza într-un fișier header (.h)

```
class Stiva {
  int *tab; //tablou care contine elementele listei
  int dim; //numărul maxim de elemente din stivă
```

```
  int index; //indexul elementului curent din lista

public:
  Stiva(int d=10); /**Constructor*/
  ~Stiva();  /**Destructor*/
  bool isEmpty(){  /**functie care spune daca stiva goala * - functie inline */
    return this->dim == this->index;
 }
  bool isFull();  /**functie care spune daca stiva e plina*/
  int pop();
  void push(int);
   void display();
};
```

b. Implementarea metodelor clasei se face în exteriorul clasei intr-un fișier (.cpp). Pentru a spune că o metodă aparține unei clase, numele metodei trebuie prefixat cu numele clasei urmat de operatorul de rezoluție (ex. void Stiva::pop())

```
Stiva::Stiva(int d) {
   cout << "Apel contructor pentru obiectul " << this << " de tip Stiva cu dimensiunea "<< d << endl;
  if (d<=0) throw "Dimensiunea stivei trebuie sa fie mai mare decat 1.";
  this->dim = d;
  this->index = 0;
  this->tab = new int[this->dim];
}
 Stiva::~Stiva(){
  cout << "Apel destructor pentru obiectul " << this << " de tip Stiva cu dimensiunea "<< this- >dim<<
endl;
  if (this->tab != nullptr) delete []tab;
 }
bool Stiva::isFull(){ … }
int Stiva::pop(){
  if (this->isEmpty()) throw "Stiva este goala nu se mai pot extrage elemente";
 --this->index;
 int v = this->tab[index];
 this->tab[index] = 0; //elementul sters il inlocuim cu zero
 return v;
 }
 void Stiva::push(int el){ … }
 void Stiva::display(){ … }
```

c. Crearea de obiecte de tip Stivă (instanțe ale tipului Stivă) se va realiza în funcția main().

```
int main() {
  Stiva s; //declaratie unui obiect de tip stiva
 s.push(7);
```

```
  s.display();
….
   Stiva *s1 = new Stiva(5);
 ….
  delete s1;
  return 0;
}
```

d. Pentru a rezolva problema putem defini un constructor de copiere care are următorul prototip: Stiva(const Stiva &);

II. Scrieți o clasă Linie care are următoarele atribute coordonatele punctului de unde începe linia și coordonatele punctului unde se termină linia. Realizați un program care:
   a) Adaugă operația de desenare pe ecran a liniei
   b) Creează un tablou de linii și desenează pe ecran liniile tabloului.
   c) Creați o nouă clasă care să vă permită desenarea altei figuri.

Observație:

Desenarea liniilor se va realiza folosind biblioteca grafica openGlut. Deoarece scopul acestui curs nu este a va învăța cum sa folosiți această bibliotecă pentru a desena, pe lângă bibliotecile care sunt necesare pentru a rula openGlut o să primiți o interfață care ascunde detaliile de implementare și va oferi câteva primitive simple care vor ajuta în procesul de desenare. Pentru utilizarea acestei biblioteci trebuie să utilizați următorii pași:

i. Crearea unui nou proiect de tip C++ console application și rularea (executare sa)
ii. . Downladați și desarhivați arhiva corespunzătoare sistemului vostru de operare: windows_32.zip, linux_32.zip, linux_64.zip
iii. Deschideți folosind un file explorer directorul care conține proiectul creat și copiați:
   a. Fișierele EngineGlut.h, libEngineGlut.a si libEngineGlut.def în folderul proiectului
   b. [optional pe linux] Copiati fișierele libEngineGlut.dll si freeglut.dll în folderele bin/Debug din în folderul proiectului
iv. În CodeBlocks apăsați click dreapta pe numele proiectului și selectați opțiunea „Add files …” și selectați fișierul EngineGlut.h. Adaugați fișierul atât pentru modul debug cât și pentru modul release. (Pentru a lega interfața care oferă funcțiile de desenare la proiect)
v. În CodeBlocks apăsați click dreapta pe numele proiectului și selectați opțiunea „Properties” și apoi apăsați „Build Options -> Linker settings -> Add” și selectați bibilioteca libEngineGlut.a și la întrebarea de relative path se selectează butonul NO
   !Aveti grija ca sa fie selectat numele proiectului nu doar una din variantele release au debug in tree-list-ul din partea stângă a ferestrei
vi. Înlocuiți codul din fișierul main.cpp cu următorul cod ca să vedeți ca funcționează

```
#include <iostream>
#include "EngineGlut.h"
using namespace std;
int main(int argc, char** argv) {
   initEngineGlut(argc, argv);
  EngineGlut engine;
  engine.drawLine(0, 0, 100, 100);
  cout << "Hello world!" << endl;
  cin.ignore();
  return 0;
}
```

Probleme suplimentare

1. Creaţi o clasa punct care caracterizează un punct 2D: poziţia pe x, y, culoare. Pentru clasa creata definiţi operaţiile de afişare, translatarea unui punct, schimbare a culori. Realizaţi un program de exemplificare a funcţiilor folosite.
2. Creaţi o clasa Carte care are ca atribute: autor, nume, ISBN şi preţ. Creaţi un şir de cărţi şi afişaţi cărţile din şir care au autorul egal cu un autor citit de la tastatură.
3. Creaţi clasa Bloc care are ca atribute: număr etaje, adresă, număr de apartamente, informaţia dacă este abilitat termic sau nu. Realizaţi un program care:
    a. Creează şi afişează informaţii despre obiecte de tip bloc
    b. Defineşte o funcţie care permite modificarea numărului de etaje, respectiv apartamente. Exemplificaţi utilizarea funcţiei;
    c. Să se realizeze o validare a informaţiilor introduse despre blocuri (ex. numărul de etaje nu poate fi negativ)
4. Creaţi clasa Stilou care are ca atribute: firmă producătoare, tip peniţă, preţ. Realizaţi un program care:
    a. Creează şi afişează informaţii despre obiecte de tip stilou;
    b. Defineşte o funcţie care modifică preţul unui stilou cu un procent. Exemplificaţi utilizarea funcţiei;
    c. Adăugaţi o funcţie care permite aflarea preţului produsului cu TVA şi fără TVA.
5. Creaţi clasa Floare care are ca atribute: nume, perioadă de înflorire, specie. Realizaţi un program care:
    a. Creează şi afişează informaţii despre obiecte de tip floare
    b. Defineşte o funcţie care permite adăugarea unei alte perioade de înflorire pe lângă cea existentă. Exemplificaţi utilizarea funcţiei;
    c. Mai adăugaţi 2/3 atribute clasei floare pe lângă cele enumerate mai sus

Informatii suplimentare despre partea de desenare

Daca doriți sa animați un desen puteți porni de la urmatorul exemplu care animează o linie.

```
#include <iostream>
```

```
#include "EngineGlut.h"
#include <unistd.h>
using namespace std;
int main(int argc, char** argv) {
  initEngineGlut(argc, argv);
  EngineGlut engine;
  int sleepTime = 500000;
  bool b = true;
  for(;;){
      if (b){
         engine.drawLine(0, -100, 100, -100);

      } else {
         engine.drawLine(0, 100, 100, 100);
      }
    usleep(sleepTime);
    engine.drawClear();
    b = !b;
  }
  cout << "Hello world!" << endl;
  cin.ignore();
  return 0;
}
```

Primitivele puse la dispozitie de mini biblioteca grafica sunt:

- drawPoint(x, y) – desenează un punct
- drawLine (x1, y1, x2, y2) – desează o linie
- drawSurface(x1, y1, x2, y2,x2,y2) – desenează un triunghi plin
- drawClear() – curăță (șterge) suprafața grafică