Laboratorul 1

Paradigme de Programare

Reamintire cunostiinte de baza de la C si CPP

Evaluare/Notare laborator si teme acasa

Task-urile: trebuiesc indeplinite in timpul laboratorului pentru a obtine punctajul aferent de 25 %

Tema pe acasa: trebuiesc predate in laboratorul urmator (altfel NU se mai accepta) pentru a obtine punctajul aferent de 25 %

Task 1. Deschideti MS Visual si testati (minute)

```c
#include <stdio.h>

int main(void) {
	char s[100];
	printf("Introduceti un sir de caractere: ");
	fflush(stdout);
	scanf("%s", s);
	printf("Sirul de caractere introdus este: %s", s);
	return 0;
}
```

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

int main() {
	char s[100];
	cout << "Introduceti un sir de caractere: ";
	cin >>  s;
	cout << "Sirul de caractere introdus este: " << s << endl;
	return 0;
}
```

Task 2. Creati si executati vesnicul program HelloWorld in stil C (structurat) (5 minute)

Task 3. Creati si executati/depanati in CPP o clasa HelloWorld dar cu (10 minute)

- un constructor implicit care seteaza sirul de afisat ca fiind “HelloWolrd”
- un constructor explicit care permite primirea la initalizare a unui sir e afisat
- o metoda care afiseaza mesajul
- o metoda care sterge mesajul

Task 4. Folosind pointeri void si cast-ul explicit definiti, initializati, modificati, interpretati multiplu aceasi variabila fara tip 10 minute

Task 5. Creati o ierarhie de clase formata din (20 minute)

- clasa abstracta mamifer (attribute (tip- (evil,divine), data nasterii) si metode (mananca, merge la baie, hraneste animalele)
- Se deriveaza o clasa bunica si clasele PisicaDeCartier, PisicaSiameza si Pisica egipteana

Acestea din urma mai au metoda miauna unde se aplica polimorfismul si fiecare miauna diferit

Actiunile in acest caz sunt implementate prin mesaje afisate la consola

Task 6. Analizand codul de mai jos sa fie reproiectat (extragere relatii intre clase, eliminarea referintelor la apelul zone grafice si afunctiilor asociate) astfel incat sa functioneze in mod text (cerc e mai complicat dispare) dar line si ca forma sa avem patrat si dreptunghi 50 min

Codul sursă initial

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
#include <stdlib.h>
#include <string.h>

char read buffer[1000];
char prompt[100];

class Form
{
protected: int *params;
public: virtual int read() = 0;
virtual void paint() = 0;
virtual void status(char*s) = 0;
};
class LineForm : public Form
{
public: LineForm()
        { params = new int[4];}
int read();
void paint();
void status(char *s);
};
class CircleForm : public Form
{
public: CircleForm()
  { params = new int[3];}
int read();
void paint();
void status(char *s);
};

void graph init();
```

```
Form* get command();
Form* forms[] =  {new LineForm(), new CircleForm()};
  int n forms = 2;
  void main ()
  { clrscr();
  graph init();
  sprintf(prompt, "enter command");
  Form *form;
  while(1) {
              gotoxy(1, 1);
              clreol();
              fflush(stdin);
              printf("[%s]> ", prompt);
              gets(read buffer);
              sprintf(prompt, "");
              form = get command();
              if (form)
                 {
                 form->paint();
                 form->status(prompt);
                 }
              else
                 {
                 if (strcmp(read buffer, "exit") == 0)
                     break;
                 if (strcmp(read buffer, "clear") == 0)
                     {
                      cleardevice();
                      sprintf(prompt, "enter command");
                      continue;
                     }
                 if (!strlen(prompt))
                    sprintf(prompt, "invalid command");
              }
       }
       closegraph();
  }
  void graph init()
       {
       int gd = DETECT, gm;
       initgraph(&gd, &gm, "c:\\borlandc\\bgi");
       }
  Form* get command()
       {
       for (int i=0; i<n forms; i++)
        if(forms[i]->read())
                return forms[i];
       return NULL;
       }
  int LineForm::read()
```

```
        {
        int res;
        char *s;
        res = sscanf(read buffer, "line(%d,%d,%d,%d%s", params, params+1, params+2,
        params+3, s);
        if (res == 5 && strcmp(s, ")") == 0)
          return 1;
        else
          {
          if (strcmp(read buffer, "help line") == 0)
               sprintf(prompt, "line(int x1, int y1, int x2, int y2)");
          return 0;
          }
        }
void LineForm::paint()
        {line(*params, *(params+1), *(params+2), *(params+3));}
void LineForm::status(char *s)
        {sprintf(s, "line(x1: %d, y1: %d, x2: %d, y2: %d)", *params, *(params+1), *(params+2), *(params+3));}


int CircleForm::read()
        {
         int res;
        char *s;
        res = sscanf(read buffer, "circle(%d,%d,%d%s", params, params+1, params+2, s);
         if (res == 4 && strcmp(s, ")") == 0)
          return 1;
        else
          {
          if (strcmp(read buffer, "help circle") == 0)
                        sprintf(prompt, "circle(int x, int y, int radius)"); return 0;
          }
        }
void CircleForm::paint()
        {circle(*params, *(params+1), *(params+2));}
void CircleForm::status(char *s)
        {sprintf(s, "circle(x:%d, y:%d, radius:%d)",*params, *(params+1), *(params+2));}
```

Tema pe ACASĂ

Extindeţi aplicaţia oferind suport pentru mai multe forme geometrice (pe lângă comanda de desenare în sine, nu uitaţi de opţiunea *help).*

Pe lângă cele două forme simple deja prezentate, încercaţi sa adăugaţi altele noi: triunghi, hexagon, si pentagon.