Vaje 15

Simona Perčič

16. in 17.02.1998

Shema ocenjevanja 1. domače naloge

SKUPAJ TOCK: 100

1. Module ART:(60)
   (a) pravilnost(30)
      - podprogram ART_Sub (15)
        - rokovanje različnih predznakov (5)
        - rokovanje z velikostjo polja in števila(5)
        - samo odštevanje (5)
      - podprogram ART_Add(15)
        - rokovanje različnih predznakov (5)
        - rokovanje z velikostjo polja in števila (5)
        - samo seštevanje(5)

   (b) kodiranje (20)
      - uporaba podprogramov in konstant(10)
      - komentarji, zaimki itd.(10)

   (c) programska dokumentacija(10)
      - vsebina komentarja (10)

2. Testni modul(40)
   (a) pravilnost(15)
      - uporaba 'feof' v glavni zanki(2)
      - uporaba 'scanf' za branje vrstice(2)
      - uporaba 'switch' glede na prebrani ukaz(3)
      - pravilno izpisovanje števila (3)
      - pravilni klici podprogramov (2)

   (b) kodiranje (10)
      - uporaba podprogramov in konstant (5)

- komentarji, zaimki itd.(5)

(c) programska dokumentacija (5)

- vsebina komentarjev (5)

(d) ali se program prevede in izvede(10)

Rešitve 1.domače naloge si lahko ogledate na področju:

`Z:\PREDAVANJA\POLDE\...`

Na primeru SKLAD si oglejmo kaj več o *polimorfizmu*

Na danem primeru si lahko ogledate kako izgleda *objektno programiranje* proti *polimorfizmu.*

Definicijski modul najdemo na področju

`Z:\Predavanja\Polde\code\Sklad\Sklad.h`

```
#include "../../code/bool/bool.h"

#define max_Sklad 100

class Sklad {                 //Sklad-ime razreda

private:                      //zasebno
        int skladisce [max_sklad];
        int zadnji;

public:
  Sklad();                    //KONSTRUKTOR
        /*Desc: Ustvari prazen sklad.
          Parms: /
          Result: Sklad
          Pre: /
          Post: SKL_Prazen()
        */

  ~Sklad();                   //DESTRUKTOR
        /*Desc: Unici sklad.
          Parms: /
          Result: /
          Pre: Sklad je bil ustvarjen
          Post: Sklad ne moremo vec uporabiti.
        */
```

```
  void Nalozi( int element );
        /*Desc: Element nalozi na sklad..
          Parms: element-element, ki ga nalozimo na sklad
          Result: /
          Pre: not(SKL_Poln())
          Post: Sklad´= [element|Sklad]
        */

  int Zlozi();
        /*Desc: Zlozi element s sklada.
          Parms: /
          Result: element
          Pre: (not (SKL_Prazen()) && (Sklad = [element|ostalo])
          Post: Sklad´ = ostalo
        */

  bool Prazen();
        /*Desc: Preveri, ali je sklad prazen.
          Parms: /
          Result: { if Sklad = [] then TRUE else FALSE }
          Pre: Sklad  ustvarjen
          Post: /
        */

  bool Poln();
        /*Desc: Preveri, ali je sklad poln.
          Parms: /
          Result: b if Sklad.zadnji < max_Sklad then FALSE else TRUE }
          Pre: Sklad je bil ustvarjen.
          Post: /
        */

  int Vrh();
        /*Desc: Vrne vrhni element Sklada.
          Parms: /
          Result: vrh
          Pre: not(Prazen()) && [vrh|_]
          Post: /
        */

}  //Sklad
```

Razred

Deklaracijo razreda sestavljata dva dela. Prvi del je glava razreda, ki jo sestavlja rezervna beseda *class* in ime razreda. Drugi del pa predstavlja telo razreda in je zaprt v zavitih oklepajih.V telesu razreda se nahajajo objekti(spremenljivke) in metode (funkcije).

Skrivanje podatkov

- Člani razreda, ki ki so deklarirani kot *public*, so dostopni kjer koli iz programa. Tako naj bi bile definirane le metode, ki omogocajo uporabo objektov.
- Člani razreda, ki so deklarirani kot *private*, so dosegljivi le metodam samega razreda.

Konstruktorji

- Konstruktor je metoda razreda, **ki ima isto ime kot razred**.
- Konstruktor je metoda razreda, **ki ničesar ne vrača** (pred njo ne sme stati niti `void`)!
- Razred ima **lahko več** konstruktorjev.
- Konstruktor se izvede ob **deklaraciji** razredov.

Destruktorji

- Destruktor je metoda razreda, **ki ima isto ime kot razred in dodan prvi znak „˜“.**
- Destruktor je ime razreda, **ki nič ne vrača** (pred njo ne sme stati niti `void`!) in ničesar ne sprejme kot parameter.
- Razred ima lahko **samo en destruktor**.
- Destruktor se izvede, **ko se objekt uniči**.

Izvedbeni modul najdemo na področju:

`Z:\Predavanja\Polde\code\Sklad\Sklad.cpp`

```
#include "../../code/Sklad/Sklad.h"

Sklad:: Sklad()  {
  zadnji = 0;
} //Sklad

Sklad:: ~Sklad()  {}

void Sklad:: Nalozi (int element)  {
  skladisce[zadnji] = element;
```

```
  zadnji++;
} //Nalozi

int Sklad:: Zlozi() {
  zadnji--;
  return skladisce[zadnji];
} //Zlozi

bool Sklad:: Prazen() {
  return (zadnji == max_Sklad);
} //Poln

int Sklad:: Vrh() {
  return skladisce[zadnji-1];
} //Vrh
```

Testni modul si lahko ogledati na področju

`Z:\Predavanja\Polde\test\Sklad\Testsk.cpp`

```
#include <stdio.h>
#include "../../code/Sklad/Sklad.h"


int main() {
  int element;
  Sklad s;                // s.Sklad-s tem pozenemo funkcijo Skladin
                          // s postane Sklad
  printf("Napisi stevilko, 0 predstavlja konec \n");
  scanf("%d", &element);
  while (element !=0) {
         s.Nalozi(element);
         scanf("%d", &element);
  }
  while (! S.Prazen())
         printf("%d \n", s.Zlozi());
}
```