PROGRAMSKI JEZICI 1 – K1 (22.11.2016.)

1. U prostoru imena **`test`** apisati klasu **`Point`** koja predstavlja tačku u dvodimenzionom realnom koordinatnom sistemu. Tačka treba da ima jedan konstruktor koji omogućava inicijalizovanje obe koordinate, implicitno kastovanje iz tipa **`double`** u *x* koordinatu (pri čemu se *y* koordinata postavlja na 0), te inicijalizovanje tačke na koordinatni početak. Omogućiti ispis tačke na standardni izlaz preklapanjem odgovarajućeg operatora. Podaci članovi koji predstavljaju koordinate treba da budu javni.
   1.1. Koji je drugi način za omogućavanje javnosti svih podataka članova?
   1.2. Klasa **`Point`** nije realizvana prema OOP principima. Kako je potrebno enkapsulirati podatke članove da bi to bilo ispravno izvedeno?
   1.3. Mora li operator za ispis na standardni izlazni tok biti proglašen za prijateljsku funkciju ove klase?
   1.4. Može li operator za ispis na standardni izlaz da bude implementiran kao nestatička funkcija članica?
2. U prostoru imena **`test`** napisati klasu **`BufferedList`** koja predstavlja baferisanu dinamičku kolekciju elemenata klase **`Point`**. Baferisana lista se realizuje kao predalociran niz elemenata koji se realocira (dopunjavanjem određenim brojem nultih elemenata i čuvanjem originalnog sadržaja) kada se prethodno alocirani niz popuni. Nije dozvoljeno korištenje struktura podataka iz standardne biblioteke. Baferisana lista treba da implementira sljedeće funkcionalnosti.
   - Odgovarajuće konstruktore, destruktore i operatore koji omogućavaju ispravan i konzistentan rad objekata ove klase (izbjegavaju curenje memorije, nedozvoljen pristup memoriji, te optimizuju prebacivanje privremenih objekata).
   - Konstruktor koji prihvata jedan parametar neoznačenog cjelobrojnog tipa koji predstavlja inicijalni kapacitet liste. Inicijalni kapacitet liste je ujedno i broj članova za koji će se povećavati kapacitet liste pri realokaciji. Ovaj podatak član treba da bude konstantan. Lista treba da čuva podatak o trenutnom kapacitetu i trenutnom broju elemanata. Onemogućiti implicitnu konverziju iz neoznačenog cjelobrojnog tipa u baferisanu listu. Podrazumijevani inicijalni kapacitet liste treba da bude 16 elemenata i treba da bude definisan kao konstanta u heder fajlu.
   - Operator indeksiranja koji omogućava pristup i izmjenu *i*-tog elementa liste (ukoliko je indeks izvan opsega, vratiti nulti element).
   - Metoda **`add(...)`** koja dodaje tačku u listu. Ukoliko dodavanje prelazi trenutno alociran kapacitet liste, ova metoda treba da pozove pomoćnu metodu **`void realloc()`** koja treba da realocira niz uvećavajući njegov kapacitet za vrijednost inicijalnog kapaciteta.
   - Operator **`()`** koji prihvata lambda funkciju za filtriranje tačaka i vraća novu baferisanu listu koja sadrži samo one tačke koje su zadovoljile uslov filtriranja definisan proslijeđenom lambda funkcijom (odnosno, lista koju vraća operator sadrži samo elemente za koje je uslov filtriranja bio istinit).
   - Operator za ispis na standardni izlaz koji ispisuje svaku tačku liste u zasebnoj liniji.
3. U glavnom programu potrebno je uraditi sljedeće korake (tačno onako kako su navedeni).
   - Instancirati baferisanu listu.
   - Alocirati dinamički niz od 10 tačaka i inicijalizovati ih redom na (0,0), (1,1), (2,2) itd., dodati te tačke u listu, a potom dealocirati dinamički niz.
   - Alocirati statički niz od 10 tačaka i inicijalizovati prve četiri tačke na (1,2), (1,3), (1,4) i (1,5), a potom sve tačke iz tog niza dodati u listu korištenjem for petlje sa for-each sintaksom definisanom C++11 standardom.
   - Na standardni izlaz ispisati sve tačke iz liste koje se nalaze na pozitivnoj strani x ose, pozivom odgovarajućih operatora nad klasom **`BufferedList`**.
   3.2. Za ovako napisan program, koliko će tačaka biti ispisano na standardni izlaz?
   3.3. U kodu napisanog programa, označiti komentarom sadržaja **`//MV`** svaku liniju u kojoj se aktivira konstruktor sa pomjeranjem klase **`BufferedList`**.
   3.4. Dat je sljedeći segment koda. Ukoliko se kod nalazi iznad prazne definicije funkcije **`main`**, šta će biti ispisano pri izvršavanju takvog programa?

```
#include <iostream>
struct A {
    A() { std::cout << "A" << ++x; }
    A(int x) : A() { std::cout << x; }
    ~A() { std::cout << "D"; }
    static int x;
};
int A::x = 0;
struct B {
    A a, aa, aaa;
    B() : aa(1), a(2) { std::cout << "B" << std::endl; }
    ~B() { std::cout << "B" << A::x; }
};
B beta;
```