KLASE - KONSTRUKTORI I DESTRUKTORI

1. Sastaviti klasu Datum koja od privatnih podataka ima dan, mesec i godinu. Obezbediti getere za data polja, kao i metodu kojom se datum upoređuje sa drugim datumom. U glavnom programu učitati listu datuma, a zatim na izlazu ispisti dobijenu listu, kao i najkasniji datum.

```
#pragma once
#include <iostream>
using namespace std;

class Datum
{
   int d, m, g;
   static int dani[2][12];
public:
   static bool moze(int d, int m, int g);
   Datum(int, int, int);
   ~Datum();
   int dan() const { return d; }
   int mes() const { return m; }
   int god() const { return g; }
   int uporedi(const Datum& dat) const;
   bool citaj();
   void pisi() const;
};
```

Datum/Datum.h

```
#include "Datum.h"

int Datum::dani[2][12] = {
   { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 },
   { 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 }
};

bool Datum::moze(int d, int m, int g)
{
   return (g > 0) && (m > 0) && (m <= 12) && (d > 0) && (d <= dani[g % 4 == 0][m - 1]);
}

Datum::Datum(int dd = 7, int mm = 11, int gg = 2015)
{
   if (!moze(dd, mm, gg)) exit(1);
   d = dd; m = mm; g = gg; // mora ovde da se setuju vrednosti atributa!
}

int Datum::uporedi(const Datum& dat) const
```

```
{
   if (g != dat.g) return g - dat.g;
   if (m != dat.m) return m - dat.m;
   return d - dat.d;
}

bool Datum::citaj()
{
   int d, m, g;
   cin >> d >> m >> g;
   if (!moze(d, m, g)) return false;
   *this = Datum(d, m, g);
   return true;
}

void Datum::pisi() const
{
   cout << d << '.' << m << '.' << g << '.';
}

Datum::~Datum()
{
}
```

Datum/Datum.cpp

```
#include "Datum.h"
#include <vector>

int main()
{
   int n;
   cout << "Broj datuma? ";
   cin >> n;
   while (n > 0)
   {
      vector<Datum> datumi;
      datumi.reserve(n);

      for (int i = 0; i < n; )
      {
         cout << i + 1 << ". datum? ";
         int a, b, c;
         cin >> a >> b >> c;
         if (Datum::moze(a, b, c))
         {
            datumi.push_back(Datum(a, b, c));
            i++;
         }
         else
         {
            cout << "Nevalidan datum, ucitajte ponovo! " << endl;
         }
      }

      int imin = 0;
      for (int i = 1; i < n; i++)
      {
         if (datumi[imin].uporedi(datumi[i]) < 0)
```

```
34          {
35              imin = i;
36          }
37      }
38
39      cout << "Najmladji datum: ";
40      datumi[imin].pisi();
41
42      cout << "Broj datuma? ";
43      cin >> n;
44  }
45  return 0;
46 }
```

Datum/Source.cpp

2. Realizovati klasu Tekst koja će predstavljati dinamičke znakovne nizove nad kojima je moguće izvršiti sledeće operacije: izračunavanje dužine znakovnog niza, čitanje i zamenu zadatog karaktera u nizu. Obezbediti odgovarajuće konstruktore i destruktore za datu klasu.

```
1  #pragma once
2  #include <string>
3  #include <iostream>
4  using namespace std;
5
6  class Tekst
7  {
8      char *niz;
9  public:
10     Tekst();
11     Tekst(char *);
12     Tekst(const Tekst&);
13     ~Tekst();
14
15     int tLength() const;
16     char tRead(int) const;
17     void tWrite(int, char) const;
18 };
```

Tekst/Tekst.h

```
1  #include "Tekst.h"
2
3  Tekst::Tekst(char *t)
4  {
5      int n = strlen(t);
6      this->niz = new char[n + 1];
7      strcpy_s(this->niz, n + 1, t);
8  }
9
10 Tekst::Tekst(const Tekst& t)
11 {
12     int n = strlen(t.niz);
13     this->niz = new char[n + 1]; // rezervacija memorije
14     strcpy_s(this->niz, n + 1, t.niz);
15 }
```

```
16
17 Tekst::Tekst()
18 {
19    this->niz = 0;
20 }
21
22
23 Tekst::~Tekst()
24 {
25    delete[] this->niz;
26    this->niz = 0;
27 }
28
29 int Tekst::tLength() const
30 {
31    return strlen(this->niz);
32 }
33
34 char Tekst::tRead(int i) const
35 {
36    return this->niz[i];
37 }
38
39 void Tekst::tWrite(int i, char c) const
40 {
41    niz[i] = c;
42 }
```

Tekst/Tekst.cpp

```
1 #include "Tekst.h"
2
3 int main()
4 {
5    Tekst t1("pozdrav svima");
6    Tekst t2(t1);
7    for (int i = 0; i < t1.tLength(); i++)
8    {
9       cout << t1.tRead(i);
10   }
11   cout << endl;
12
13   t2.tWrite(0, 'w');
14   for (int i = 0; i < t2.tLength(); i++)
15   {
16      cout << t2.tRead(i);
17   }
18   cout << endl;
19
20   /*
21   Tekst pozdrav("Pozdrav svima");
22   Tekst b;
23
24   cout << "pozdrav = ";
25   for (int i = 0; i < pozdrav.tLength(); i++)
26   {
27      cout << pozdrav.tRead(i);
28   }
29   cout << endl;
```

```
30
31     cout << "mesto karaktera? ";
32     int j;
33     cin >> j;
34
35     cout << "koji karakter? ";
36     char c;
37     cin >> c;
38
39     pozdrav.tWrite(j, c);
40     cout << "novi pozdrav je ";
41     for (int i = 0; i < pozdrav.tLength(); i++)
42     {
43        cout << pozdrav.tRead(i);
44     }
45     cout << endl;
46     */
47     return 0;
48 }
```

Tekst/Source.cpp

3. Realizovati klasu red koja predstavlja niz celih brojeva proizvoljne dužine, a pritom klasa sadrži i podatak o dužini reda. Omogućiti da indeks prvog člana može biti različit od nule ukoliko korisnik to zada. Omogućiti pristup članovima reda za upis i izmenu podataka. Napisati glavni proram u kojem će se unositi odgovarajući red i na izlazu davati suma članova tog reda koji je sastavljen od kvadrata elemenata reda koji je korisnik uneo.

```
1  #pragma once
2  #include <iostream>
3  using namespace std;
4
5  class Niz
6  {
7     int *p; // pok. na pocetak niza celih brojeva
8     int num; // duzina niza
9     const int i1; // indeks pocetnog elementa
10 public:
11    Niz(int first_index = 1, int number = 0);
12    Niz(const Niz&); // konstruktor kopije
13    ~Niz();
14
15    int first() const { return i1; }
16    int length() const { return num; }
17    int last() const { return i1 + num - 1; }
18    int read(int index) const { return p[index - i1]; }
19    void change(int index, int value)
20    {
21       p[index - i1] = value;
22    }
23
24    int sum() const;
25    void square();
26 };
```

Niz/Niz.h

```cpp
#include "Niz.h"

Niz::Niz(int first_index, int number)
   : i1(first_index), num(number)
{
   p = new int[num];
}

Niz::Niz(const Niz& a)
   : i1(a.i1), num(a.num)
{
   p = new int[num];
   for (int i = 0; i < num; i++)
   {
      p[i] = a.p[i];
   }
}

Niz::~Niz()
{
   delete[] p;
   p = 0;
}

int Niz::sum() const
{
   int s = 0;
   for (int i = first(); i <= last(); i++)
   {
      s += read(i);
   }
   return s;
}

void Niz::square()
{
   for (int i = first(); i <= last(); i++)
   {
      int x = read(i);
      change(i, x * x);
   }
}
```

Niz/Niz.cpp

```cpp
#include "Niz.h"

int main()
{
   int n, i, x;
   cout << "Koliko elemenata ima niz?" << endl;
   cin >> n;

   cout << "Od kojeg broja pocinje indeksiranje niza?" << endl;
   cin >> i;

   cout << "Unesite elemente niza:" << endl;
   Niz a1(i, n);
```

```
14      for (int j = i; j < n + i; j++)
15      {
16          cin >> x;
17          a1.change(j, x);
18      }
19
20      Niz a2(a1);
21      a2.square();
22      cout << "Suma clanova prvog niza je: " << a1.sum() << endl;
23
24      cout << "Niz dobijen kvadriranjem clanova prvog niza je: " << endl;
25      for (int j = a2.first(); j <= a2.last(); j++)
26      {
27          cout << a2.read(j) << " ";
28      }
29      return 0;
30  }
```

Niz/Source.cpp