I PARCIJALNI ISPIT IZ PREDMETA "TEHNIKE PROGRAMIRANJA" (GRUPA A)

Napomena: Razumije se da sva prikazana rješenja (osim za Zadatak 1) predstavljaju samo jedno od mnoštva mogućih ispravnih rješenja. Prikazano rješenje je tipično najkraće od svih mogućih rješenja.

Zadatak 1 (5 poena)

Prikažite *tačan izgled ekrana* na kraju izvršavanja ovog C++ programa, uz kratko obrazloženje zbog čega su rezultati onakvi kakvi jesu (program će ukupno ispisati 5 redova teksta, pri čemu svaki posve ispravno otkriveni red nosi po 1 poen). Oprez: bitan je svaki razmak, kao i prelasci u nove redove. Radi jasnoće, razmake prikažite kao kvadratiće.

```
#include <iostream>
#include <iomanip>
#include <complex>
using namespace std;
const char s(' ');
void F1(int &a, int b, int &c) {
  a = b + c; b = a + c; c = a + b;
  cout << a << s << b << s << c << s;
}
void F2(int a, int &b, int c) {
  a = b + c; b = a + c; c = a + b;
  cout << a << s << b << s << c << s;
}
int &F3(double (*f)(double), int *p) {
  p[1] = int(f(*p)); return *(--p);
}
int main() {
  int x(4); F1(x, x, x); cout << x << endl;
  x = 4; F2(x, x, x); cout << x << endl;
  int a1(5); int a2(a1); const int a3(a1);
  int &a4(a1); const int &a5(a1); const int &a6(a1 + 0);
  a1 += 2; cout << a1 << a2 << a3 << a4 << a5 << a6 << endl;
  double c1(9); complex<double> c2(c1); complex<double> c3(-c2);
  cout << sqrt(c1) << setw(7) << sqrt(c2) << " " << sqrt(c3) << endl;
  int niz[5] = {3, 7, 4, 5, 2};
  F3(sqrt, niz + 2)++;
  for(int i = 0; i < 5; i++) cout << niz[i] << " ";
  return 0;
}
```

Rješenje:

```
24□16□24□24
8□12□20□12
755775
3□□(3,0)□(0,3)
3□8□4□2□2
```

Zadatak 2 (1,5 poen)

Napišite funkciju sa jednim cjelobrojnim parametrom koja kao rezultat vraća logičku vrijednost "tačno" ili "netačno" u ovisnosti od toga da li su sve cifre broja proslijeđenog kao parametar parne ili ne. Napišite i kratki isječak programa u kojem biste demonstrirali kako se napisana funkcija može iskoristiti da se ispiše da li broj unesen sa tastature ima sve cifre parne ili ne.

Rješenje:

```
bool DaLiSuSveCifreParne(int n) {
  while(n != 0) {
    if(n % 2 != 0) return false;
    n /= 10;
  }
  return true;
}

...

int broj;
cin >> broj;
if(DaLiSuSveCifreParne(broj)) cout << "Sve cifre unesenog broja su parne";
else cout << "Neke cifre unesenog broja nisu parne";
```

Zadatak 3 (2,5 poena)

Napišite funkciju koja kao parametar prima vektor cijelih brojeva, a koja kao rezultat vraća novi vektor cijelih brojeva čiji su elementi sume svih djelilaca odgovarajućih elemenata prvog vektora (npr. ako je treći element prvog vektora 18, treći element novog vektora treba biti 39, jer djelioci broja 18 glase 1, 2, 3, 6, 9 i 18, a vrijedi 1+2+3+6+9+18 = 39).

Napomena: U svim zadacima gdje nije drugačije rečeno, napišite *samo funkciju*, i ništa više.

Rješenje:

```
vector<int> SumeDjelilaca(const vector<int> &v) {
  vector<int> v1;
  for(int i = 0; i < v.size(); v++) {
    int suma(0);
    for(int j = 1; j <= v[i]; j++)
      if(v[i] % j == 0) suma += j;

    v1.push_back(suma);
  }
  return v1;
}
```

Zadatak 4 (2 poena)

Napišite generičku funkciju sa 4 parametra. Prvi parametar je niz elemenata proizvoljnog tipa, za koje se pretpostavlja da se mogu međusobno upoređivati, dok je drugi parametar broj elemenata u tom nizu. Funkcija treba da u treći i četvrti parametar smjesti respektivno najmanji element niza i mjesto gdje se on nalazi u nizu (tj. njegov indeks). Ukoliko ima više najmanjih elemenata, treba smjestiti mjesto posljednjeg takvog elementa u nizu. Napišite i mali isječak programa u kojem ćete demonstrirati kako se može upotrijebiti napisana funkcija.

Rješenje:

```
template <typename UporediviTip>
void NadjiNajmanji(UporediviTip niz[], int br_elemenata, UporediviTip &min,
  int &gdje_je) {
    min = niz[0]; gdje_je = 0;
    for(int i = 1; i < br_elemenata; i++)
      if(niz[i] <= min) {
        min = niz[i]; gdje_je = i;
      }
}

...
```

```
int a[10] = {3, 5, 2, 8, 1, 6, 1, 4, 2, 3};
int min, gdje_je;
NadjiNajmanji(a, 10, min, gdje_je);
cout << "Najmanji element niza je " << min
  << " a posljednji put se javlja na poziciji " << gdje_je << endl;
```

Zadatak 5 (1,5 poen)

Napišite funkciju koja prima dva parametra "s1" i "s2" koji su dinamički stringovi (tj. objekti tipa "string"). Funkcija treba da kreira i vrati kao rezultat novi string koji se sastoji od onih i samo onih znakova stringa "s1" koji se također javljaju i u stringu "s2" (u istom poretku u kojem se nalaze i u stringu "s1"). Na primjer, ukoliko stringovi "s1" i "s2" glase respektivno "sdfgffwestz" i "szdgzvc", funkcija treba da kao rezultat vrati string "sdgsz".

Rješenje:

```
string ZajednickiZnakovi(const string &s1, const string &s2) {
  string s;
  for(int i = 0; i < s1.length(); i++) {
    bool ima_ga(false);
    for(int j = 0; j < s2.length(); j++)
      if(s1[i] == s2[j]) ima_ga = true;
    if(ima_ga) s.push_back(s1[i]);
  }
  return s;
}
```

Zadatak 6 (2,5 poena)

Napišite generičku funkciju koja u bloku elemenata između pokazivača "p1" i "p2" pronalazi element za koji funkcija "f" daje najveću moguću vrijednost i kao rezultat vraća pokazivač na taj element (ukoliko takvih elemenata ima više, funkcija treba da vrati pokazivač na prvi takav element). Parametri funkcije su "p1" i "p2" i "f" (takva funkcija ne postoji u biblioteci "algorithm", ali je potpuno "u duhu" funkcija iz ove biblioteke). Funkcija bi trebala biti zasnovana na potpunoj dedukciji tipova, tako da se umjesto pokazivača mogu koristiti i iteratori (ukoliko ne znate izvesti potpunu dedukciju, koristite parcijalnu dedukciju, samo ćete time izgubiti 1 poen). Napišite i mali isječak programa u kojem ćete demonstrirati kako biste ovu funkciju primijenili da iz nekog fiksnog niza od 5 realnih brojeva ispišete onaj element čiji je sinus najveći.

Rješenje:

```
template <typename PokazivackiTip, typename TipElemenata>
PokazivackiTip MaksimumFunkcije(PokazivackiTip p1, PokazivackiTip p2,
  TipElemenata f(TipElemenata)) {
    PokazivackiTip gdje_je(p1);
    for(; p1 != p2; p1++)
      if(f(*p1) > f(*gdje_je)) gdje_je = p1;
    return gdje_je;
}

...

double niz[5] = {3.7, 2.15, 4.741, 3.19, 5.2};
cout << "Najveći sinus ima element "
  << *MaksimumFunkcije(niz, niz + 5, sin);
```

Zadatak 7 (2,5 poena)

Pretpostavimo da želimo sortirati vektor stringova po dužini, u smislu da duži stringovi dolaze prije kraćih stringova, pri čemu ukoliko su dva stringa iste dužine, prije treba doći onaj koji dolazi prije po abecednom poretku (npr. string "pQrS" treba doći prije stringa "XyZ" jer je duži, ali zato string "aBc" treba doći ispred stringa "XyZ", jer dolazi prije po abecednom poretku). Definirajte odgovarajuću funkciju kriterija i napišite isječak programa u kojem ćete pokazati kako biste deklarirali vektor stringova, napunili ga vrijednostima unesenim sa tastature i sortirali ga u traženi poredak korištenjem funkcije "`sort`" iz biblioteke "`algorithm`" (podsjetimo se da ova funkcija prima kao parametre dva pokazivača ili iteratora koji omeđuju blok koji se sortira i opcionalno funkciju kriterija kao treći parametar).

Rješenje:

```
bool Kriterij(string s1, string s2) {
  if(s1.length() != s2.length()) return s1.length() > s2.length();
  transform(s1.begin(), s1.end(), s1.begin(), (int(*)(int))toupper);
  transform(s1.begin(), s1.end(), s1.begin(), (int(*)(int))toupper);
  return s1 < s2;
}

...

int br_elemenata;
cin >> br_elemenata;
vector<string> v;
cin.ignore(10000, '\n');
for(int i = 0; i < br_elemenata; i++) getline(cin, v[i]);
sort(v.begin(), v.end(), Kriterij);
```

Zadatak 8 (2,5 poena)

Napišite generičku funkciju koja služi za dinamičku alokaciju matrica čiji su elementi proizvoljnog tipa. Funkcija ima 4 parametra, od kojih se četvrti može izostaviti. Drugi i treći parametar su broj redova i broj kolona matrice respektivno, dok je četvrti parametar inicijalna vrijednost kojom se popunjavaju elementi matrice. Funkcija treba da izvrši dinamičku alokaciju prostora za pamćenje elemenata matrice, zatim da popuni elemente matrice zadanom inicijalnom vrijednošću i, konačno, da u prvi parametar funkcije smjesti dvojni pokazivač preko kojeg se može pristupiti elementima matrice. U slučaju da se četvrti parametar izostavi, za inicijalizaciju elemenata matrice treba koristiti odgovarajuću podrazumijevanu vrijednost za tip elemenata matrice. U slučaju da dinamička alokacija ne uspije, u prvi parametar funkcije treba smjestiti 0 (tj. nul-pokazivač). Ova funkcija ne vraća nikakvu vrijednost, niti baca izuzetke (tj. svi izuzeci koji bi se mogli pojaviti trebaju biti uhvaćeni i obrađeni unutar same funkcije). Vodite računa da u slučaju neuspješne alokacije ni u kom slučaju ne dođe do curenja memorije. Napišite i mali isječak glavnog programa (ne treba sve) u kojem ćete demonstrirati kako biste pozvali ovu funkciju za kreiranje matrice realnih brojeva formata 10×10 i testirali da li je alokacija matrice uspjela ili ne.

Rješenje:

```
template <typename TipElemenata>
void KreirajMatricu(TipElemenata **&mat, int br_redova, int br_kolona,
  TipElemenata inicijalna_vrijednost = TipElemenata()) {
    try {
      mat = new TipElemenata*[br_redova];
      for(int i = 0; i < br_redova; i++) mat[i] = 0;
      try {
        for(int i = 0; i < br_redova; i++)
          mat[i] = new TipElemenat[br_kolona];
      }
```

```
      catch(...) {
        for(int i = 0; i < br_redova; i++) delete[] mat[i];
        delete[] mat;
        throw;
      }
      for(int i = 0; i < br_redova; i++)
        for(int j = 0; j < br_kolona; j++)
          mat[i][j] = inicijalna_vrijednost;
    }
    catch(...) {
      mat = 0;
    }
}

...

double **a;
KreirajMatricu(a, 10, 10);
if(a == 0) cout << "Kreiranje nije uspjelo!\n";
```

I PARCIJALNI ISPIT IZ PREDMETA "TEHNIKE PROGRAMIRANJA" (GRUPA B)

Napomena: Razumije se da sva prikazana rješenja (osim za Zadatak 1) predstavljaju samo jedno od mnoštva mogućih ispravnih rješenja. Prikazano rješenje je tipično najkraće od svih mogućih rješenja.

Zadatak 1 (5 poena)

Prikažite *tačan izgled ekrana* na kraju izvršavanja ovog C++ programa, uz kratko obrazloženje zbog čega su rezultati onakvi kakvi jesu (program će ukupno ispisati 5 redova teksta, pri čemu svaki posve ispravno otkriveni red nosi po 1 poen). Oprez: bitan je svaki razmak, kao i prelasci u nove redove. Radi jasnoće, razmake prikažite kao kvadratiće.

```
#include <iostream>
#include <iomanip>
#include <complex>
using namespace std;
const char s(' ');
void F1(int &a, int b, int &c) {
  a = b + c; b = a + c; c = a + b;
  cout << a << s << b << s << c << s;
}
void F2(int a, int &b, int c) {
  a = b + c; b = a + c; c = a + b;
  cout << a << s << b << s << c << s;
}
int &F3(double (*f)(double), int *p) {
  p[1] = int(f(*p)); return *(--p);
}
int main() {
  int x(7); F1(x, x, x); cout << x << endl;
  x = 3; F2(x, x, x); cout << x << endl;
  int a1(9); int a2(a1); const int a3(a1);
  int &a4(a1); const int &a5(a1); const int &a6(a1 + 0);
  a1 -= 3; cout << a1 << a2 << a3 << a4 << a5 << a6 << endl;
  double c1(4); complex<double> c2(c1); complex<double> c3(-c2);
  cout << sqrt(c1) << setw(7) << sqrt(c2) << " " << sqrt(c3) << endl;
  int niz[5] = {3, 1, 4, 9, 2};
  F3(sqrt, niz + 3)++;
  for(int i = 0; i < 5; i++) cout << niz[i] << " ";
  return 0;
}
```

Rješenje:

```
42□28□42□42
6□9□15□9
699669
2□□(2,0)□(0,2)
3□1□5□9□3
```

Zadatak 2 (1,5 poen)

Napišite funkciju sa jednim cjelobrojnim parametrom koja kao rezultat vraća logičku vrijednost "tačno" ili "netačno" u ovisnosti od toga da li su sve cifre broja proslijeđenog kao parametar manje od 5 ili ne. Napišite i kratki isječak programa u kojem biste demonstrirali kako se napisana funkcija može iskoristiti da se ispiše da li broj unesen sa tastature ima sve cifre manje od 5 ili ne.

Rješenje:

```
bool DaLiSuSveCifreManjeOd5(int n) {
  while(n != 0) {
    if(n % 10 >= 5) return false;
    n /= 10;
  }
  return true;
}

...

int broj;
cin >> broj;
if(DaLiSuSveCifreManjeOd5(broj))
  cout << "Sve cifre unesenog broja su manje od 5";
else cout << "Neke cifre unesenog broja su veće od 5";
```

Zadatak 3 (2,5 poen)

Napišite funkciju koja kao parametar prima vektor cijelih brojeva, a koja kao rezultat vraća novi vektor cijelih brojeva čiji su elementi broj svih djelilaca odgovarajućih elemenata prvog vektora (npr. ako je treći element prvog vektora 18, treći element novog vektora treba biti 6, jer broja 18 ima 6 djelioca, koji glase 1, 2, 3, 6, 9 i 18).

Napomena: U svim zadacima gdje nije drugačije rečeno, napišite *samo funkciju*, i ništa više.

Rješenje:

```
vector<int> BrojeviDjelilaca(const vector<int> &v) {
  vector<int> v1;
  for(int i = 0; i < v.size(); v++) {
    int brojac(0);
    for(int j = 1; j <= v[i]; j++)
      if(v[i] % j == 0) brojac++;

    v1.push_back(brojac);
  }
  return v1;
}
```

Zadatak 4 (2 poena)

Napišite generičku funkciju sa 4 parametra. Prvi parametar je niz elemenata proizvoljnog tipa, za koje se pretpostavlja da se mogu međusobno upoređivati, dok je drugi parametar broj elemenata u tom nizu. Funkcija treba da u treći i četvrti parametar smjesti respektivno najveći element niza i mjesto gdje se on nalazi u nizu (tj. njegov indeks). Ukoliko ima više najvećih elemenata, treba smjestiti mjesto prvog takvog elementa u nizu. Napišite i mali isječak programa u kojem ćete demonstrirati kako se može upotrijebiti napisana funkcija.

Rješenje:

```
template <typename UporediviTip>
void NadjiNajveci(UporediviTip niz[], int br_elemenata, UporediviTip &max,
  int &gdje_je) {
    max = niz[0]; gdje_je = 0;
    for(int i = 1; i < br_elemenata; i++)
      if(niz[i] > max) {
        max = niz[i]; gdje_je = i;
      }
}

...
```

```
int a[10] = {3, 5, 2, 8, 1, 6, 1, 4, 2, 3};
int max, gdje_je;
NadjiNajveci(a, 10, max, gdje_je);
cout << "Najveći element niza je " << max
  << " a prvi put se javlja na poziciji " << gdje_je << endl;
```

Zadatak 5 (2 poena)

Napišite funkciju koja prima dva parametra "s1" i "s2" koji su dinamički stringovi (tj. objekti tipa "string"). Funkcija treba da kreira i vrati kao rezultat novi string koji se sastoji od onih i samo onih znakova stringa "s1" koji se ne javljaju u stringu "s2" (u istom poretku u kojem se nalaze i u stringu "s1"). Na primjer, ukoliko stringovi "s1" i "s2" glase respektivno "sdfgffwestz" i "szdgzvc", funkcija treba da kao rezultat vrati string "fffwet".

Rješenje:

```
string ZajednickiZnakovi(const string &s1, const string &s2) {
  string s;
  for(int i = 0; i < s1.length(); i++) {
    bool nema_ga(true);
    for(int j = 0; j < s2.length(); j++)
      if(s1[i] == s2[j]) nema_ga = false;
    if(nema_ga) s.push_back(s1[i]);
  }
  return s;
}
```

Zadatak 6 (2,5 poena)

Napišite generičku funkciju koja u bloku elemenata između pokazivača "p1" i "p2" pronalazi element za koji funkcija "f" daje najmanju moguću vrijednost i kao rezultat vraća pokazivač na taj element (ukoliko takvih elemenata ima više, funkcija treba da vrati pokazivač na posljednji takav element). Parametri funkcije su "p1", "p2" i "f" (takva funkcija ne postoji u biblioteci "algorithm", ali je potpuno "u duhu" funkcija iz ove biblioteke). Funkcija bi trebala biti zasnovana na potpunoj dedukciji tipova, tako da se umjesto pokazivača mogu koristiti i iteratori (ukoliko ne znate izvesti potpunu dedukciju, koristite parcijalnu dedukciju, samo ćete time izgubiti 1 poen). Napišite i mali isječak programa u kojem ćete demonstrirati kako biste ovu funkciju primijenili da iz nekog fiksnog niza od 6 realnih brojeva ispišete onaj element čiji je kosinus najmanji.

Rješenje:

```
template <typename PokazivackiTip, typename TipElemenata>
PokazivackiTip MinimumFunkcije(PokazivackiTip p1, PokazivackiTip p2,
  TipElemenata f(TipElemenata)) {
    PokazivackiTip gdje_je(p1);
    for(; p1 != p2; p1++)
      if(f(*p1) <= f(*gdje_je)) gdje_je = p1;
    return gdje_je;
}

...

double niz[6] = {3.7, 2.15, 4.741, 3.19, 5.2, 4.17};
cout << "Najmanji kosinus ima element "
  << *MinimumFunkcije(niz, niz + 6, cos);
```

Zadatak 7 (2 poena)

Pretpostavimo da želimo sortirati vektor stringova po dužini, u smislu da kraći stringovi dolaze prije dužih stringova, pri čemu ukoliko su dva stringa iste dužine, prije treba doći onaj koji dolazi prije po abecednom poretku (npr. string "XyZ" treba doći prije stringa "pQrS" jer je kraći, ali zato string "aBc" treba doći ispred stringa "XyZ", jer dolazi prije po abecednom poretku). Definirajte odgovarajuću funkciju kriterija i napišite isječak programa u kojem ćete pokazati kako biste deklarirali vektor stringova, napunili ga vrijednostima unesenim sa tastature i sortirali ga u traženi poredak korištenjem funkcije "`sort`" iz biblioteke "`algorithm`" (podsjetimo se da ova funkcija prima kao parametre dva pokazivača ili iteratora koji omeđuju blok koji se sortira i opcionalno funkciju kriterija kao treći parametar).

Rješenje:

```
bool Kriterij(string s1, string s2) {
  if(s1.length() != s2.length()) return s1.length() < s2.length();
  transform(s1.begin(), s1.end(), s1.begin(), (int(*)(int))toupper);
  transform(s1.begin(), s1.end(), s1.begin(), (int(*)(int))toupper);
  return s1 < s2;
}

...

int br_elemenata;
cin >> br_elemenata;
vector<string> v;
cin.ignore(10000, '\n');
for(int i = 0; i < br_elemenata; i++) getline(cin, v[i]);
sort(v.begin(), v.end(), Kriterij);
```

Zadatak 8 (3 poena)

Napišite generičku funkciju koja služi za dinamičku alokaciju matrica čiji su elementi proizvoljnog tipa. Funkcija ima 4 parametra, od kojih se četvrti može izostaviti. Drugi i treći parametar su broj redova i broj kolona matrice respektivno, dok je četvrti parametar inicijalna vrijednost kojom se popunjavaju elementi matrice. Funkcija treba da izvrši dinamičku alokaciju prostora za pamćenje elemenata matrice, zatim da popuni elemente matrice zadanom inicijalnom vrijednošću i, konačno, da u prvi parametar funkcije smjesti dvojni pokazivač preko kojeg se može pristupiti elementima matrice. U slučaju da se četvrti parametar izostavi, za inicijalizaciju elemenata matrice treba koristiti odgovarajuću podrazumijevanu vrijednost za tip elemenata matrice. U slučaju da dinamička alokacija ne uspije, u prvi parametar funkcije treba smjestiti 0 (tj. nul-pokazivač). Ova funkcija ne vraća nikakvu vrijednost, niti baca izuzetke (tj. svi izuzeci koji bi se mogli pojaviti trebaju biti uhvaćeni i obrađeni unutar same funkcije). Vodite računa da u slučaju neuspješne alokacije ni u kom slučaju ne dođe do curenja memorije. Napišite i mali isječak glavnog programa (ne treba sve) u kojem ćete demonstrirati kako biste pozvali ovu funkciju za kreiranje matrice realnih brojeva formata 10×10 i testirali da li je alokacija matrice uspjela ili ne.

Rješenje:

```
template <typename TipElemenata>
void KreirajMatricu(TipElemenata **&mat, int br_redova, int br_kolona,
  TipElemenata inicijalna_vrijednost = TipElemenata()) {
    try {
      mat = new TipElemenata*[br_redova];
      for(int i = 0; i < br_redova; i++) mat[i] = 0;
      try {
        for(int i = 0; i < br_redova; i++)
          mat[i] = new TipElemenat[br_kolona];
      }
```

```
      catch(...) {
        for(int i = 0; i < br_redova; i++) delete[] mat[i];
        delete[] mat;
        throw;
      }
      for(int i = 0; i < br_redova; i++)
        for(int j = 0; j < br_kolona; j++)
          mat[i][j] = inicijalna_vrijednost;
    }
    catch(...) {
      mat = 0;
    }
}

...

double **a;
KreirajMatricu(a, 10, 10);
if(a == 0) cout << "Kreiranje nije uspjelo!\n";
```