Bilten rešenja – kontrolni zadatak, grupa A

1.

2. Dostupne instalacione verzije razvojnog okruženja sa setup datotekom mogu biti:
✓ Bez kompajlera (codeblocks-20.03-setup.exe)
✓ Sa kompajlerom (codeblocks-20.03mingw-setup.exe)
✓ Prenosiva verzija (codeblocks-20.03mingw-nosetup.zip)

Osim prevlačenja instalacionih verzija sa setup datotekom (kao najčešći odabir pri instalaciji na operativnom sistemu Windows), mogu se prevući i verzije koje daju više fleksibilnosti tokom instalacije, ali zahtevaju i više stručnosti pri izgradnji konačne verzije IDE spremnog za instalaciju.

Tokom instalacije, može se odabrati tip instalacije
✓Custom
✓Editor
✓Minimal
✓Standard
✓Full

Opcija pedantic pri kompajliranju govori kompajleru da se striktno pridržava ANSI standarda, odbacujući bilo koji kod koji nije usklađen sa standardom uz prikaz upozorenja i poruka o grešci (warnings, errors).

3.

Postfiksno uvecanje ++	Prefiksno uvecanje ++a
int a=5; b=a++;	int a=5; b=++a;
EFEKAT: b=a; a=a+1; tj. b=5; a=6;	EFEKAT: a=a+1; b=a; tj. a=6; b=6;

Analogno za operator --

Izraz ++x je za razliku od x++ lvrednost, te zbog toga izraz x++++++ nije ispravno kodiran, jer će taj izraz se raščlaniti na izraz ((x++)++)++ (ali izraz x++ nije lvrednost).

Izraz ++++++x je ispravan (uvećanje promenljive x za 3), jer je ++x lvrednost koja se raščlanjuje u fragmente: ++(++(++x))

4. Znak za prelaz u novi red se ne kodira isto u svim operativnim sistemima
Pod DOS/Windows oznaka za kraj reda se zapisuje sa dva karaktera (CR LF - Carriage return Line feed), 0xD 0xA tj. 13 10 – istorijski razlozi (stari štampači)

Unix koristi samo karakter LF tj. 0xA
MacOS koristi samo karakter CR tj. 0XD

Programski jezik C obezbeđuje dve prelazne sekvence \n, \r
Programski jezik C++ koristi manipulator std::endl za ispis novog reda.

https://en.wikipedia.org/wiki/Newline

5.
Postoje 2 supersrećna broja dužine 1 (7, 8). Postoje 4 supersrećna broja dužine 2. To su 77, 78, 87, 88. Postoji 8 supersrećnih brojeva dužine 3. To su 777, 778, 787, 788, 877, 878, 887, 888. Dakle, kad god se dužina (broj cifara) u srećnim brojevima poveća za 1, onda se ukupan broj brojeva te dužine udvostruči. Dakle, za ukupan broj brojeva čiji broj cifara ne premašuje n, moramo redom sabirati sledeće brojeve:Let's sum up them. $2^1 = 2$, $2^1 + 2^2 = 2 + 4 = 6$, $2^1 + 2^2 + 2^3 = 2 + 4 + 8 = 14$, $2^1 + 2^2 + 2^3 + 2^4 = 2 + 4 + 8 + 16 = 30$. Može se primetiti da ukupna suma je jednaka $2^{n+1} - 2$.

U implementaciji koristimo 64-bitni celobrojni type (long long in C++).

```
#include <iostream>
#include <cmath>

using namespace std;

int main()
{
long long n;
cin>>n;
long long x= (long long) pow(2,n+1);
cout<<x-2;
}
```

6.

7.

Rešenje

Za osvajanje delimičnih poena, bilo je dovoljno obaviti celobrojno deljenje sa 95 i množenje sa 100. Ali za osvajanje svih poena, bilo je potrebno pažljivo izvršiti skaliranje uz klasifikaciju dobijene realne vrednosti na osnovu pripadnosti disjunktnim intervalima.

Ako je p broj poena pre skaliranja, a p′ broj poena nakon skaliranja, p′ se može izračunati iz proporcije p:p′=95:100
tj. $p' = p/95 \cdot 100$. Ovaj broj je potrebno zaokružiti na najbliži broj čija je jedina decimala 0 ili 5. Brojevi sa takvim decimalnim zapisom su oni koji predstavljaju ceo broj polovina. Broj polovina u broju p′ je $p' \cdot 2$ i to ne mora biti ceo broj, ali se može zaokružiti na najbliži ceo broj. Kada se dobije ceo broj polovina, tražena vrednost broja se može dobiti deljenjem tog broja polovina sa 2.

Ako dopustimo korišćenje realnih brojeva zaokruživanje se može vršiti bibliotečkom funkcijom round (vodeći pri tom računa da zbog nemogućnosti potpuno preciznog zapisa nekih brojeva može doći do grešaka). Pošto su brojevi poena celi brojevi, postoji mogućnost je da se zaokruživanje vrši korišćenjem tehnika celobrojnog deljenja.
Dakle, količnik a/b se može zaokružiti na najbliži ceo broj korišćenjem izraza $\lfloor a+\lfloor b/2 \rfloor /b \rfloor = (a+b \text{ div } 2) \text{div } b$, gde div je oznaka za operaciju celobrojnog deljenja tj. izraza (a + b/2) / b u C ili C++.

Još jedan način da se zadatak reši je da se posmatraju decimale broja p'. Njih je moguće odrediti tako što se odredi vrednost izraza $p' - \lfloor p' \rfloor$.
Ako je vrednost tog izraza u intervalu [0,0.25) tj. ako je ta vrednost veća ili jednaka nuli, a strogo manja od 0.25, tada se vrednost treba zaokružiti na $\lfloor p' \rfloor$, ako je u intervalu [0.25,0.75), tada se vrednost treba zaokružiti na $\lfloor p' \rfloor + 0.5$, a ako je u intervalu [0.75,1) tada se vrednost treba zaokružiti na $\lfloor p' \rfloor + 1$.
Ovakva klasifikacija na osnovu pripadnosti disjunktnim intervalima može se npr. realizovati grananjem (ternarnim operatorom ili ugnježđenim, korišćenjem konstrukcije else-if).

1. način

```
#include <iostream>
#include <iomanip>
#include <cmath>

using namespace std;

int main() {
  int poeni;
  cin >> poeni;
  double skalirani_nezaokruzeni = (poeni / 95.0) * 100.0;
  double skalirani_zaokruzeni = round(skalirani_nezaokruzeni * 2.0) / 2.0;
  cout << fixed << showpoint << setprecision(1)
       << skalirani_zaokruzeni << endl;
}
```

2. način

```
#include <iostream>
#include <iomanip>

using namespace std;

int roundfrac(int a, int b) {
  return (a + b/2) / b;
}

int main() {
  int poeni;
  cin >> poeni;
  double skalirani_zaokruzeni = roundfrac(poeni * 100 * 2, 95) / 2.0;
  cout << fixed << showpoint << setprecision(1)
       << skalirani_zaokruzeni << endl;
```

```
}
```

3. način

```
#include <iostream>
#include <iomanip>
#include <cmath>

using namespace std;

int main() {
  int poeni;
  cin >> poeni;
  double skalirani_nezaokruzeni = (poeni / 95.0) * 100.0;
  double skalirani_ceo_deo = floor(skalirani_nezaokruzeni);
  double skalirani_razlomljen_deo = skalirani_nezaokruzeni - skalirani_ceo_deo;
  double skalirani_zaokruzeni;
  if (skalirani_razlomljen_deo < 0.25)
    skalirani_zaokruzeni = skalirani_ceo_deo;
  else if (skalirani_razlomljen_deo < 0.75)
    skalirani_zaokruzeni = skalirani_ceo_deo + 0.5;
  else
    skalirani_zaokruzeni = skalirani_ceo_deo + 1.0;
  cout << fixed << showpoint << setprecision(1)
       << skalirani_zaokruzeni << endl;
}
```

8.

Треба да проверимо да ли је први играч победио другог, а затим и да ли
је други играч победио првог (проверавамо освојене гемове играча).

Потребна је и провера исправности резултата.
Нико не може да има више од 7 освојених гемова, тако да ако је
већи број освојених гемова већи од 7, онда је резултат неисправан. Ако је већи број освојених гемова једнак 7 код играча, онда је код другог играча број освојених гемова или 6 или 5 .
У супротном су оба играча освојила највише 6 гемова и у том случају
је резултат исправан ако важи да мањи је освојио највише 4 гема
(тада је сет завршен), а ако је освојио 5 или 6 онда се још игра.

Пошто победа првог или другог играча осигурава да је резултат
исправан, исправност треба проверавати и ако установимо
да нико још није победио.

```
#include <iostream>

using namespace std;
```

```
int main() {
  int prvi, drugi;
  cin >> prvi >> drugi;

  if (prvi > 7 || drugi > 7 ||
      (prvi == 7 && drugi < 5) || (drugi == 7 && prvi < 5))
    cout << "neispravno" << endl;
  else if (prvi > drugi && ((prvi == 6 && drugi < 5) || prvi == 7))
    cout << "pobedio prvi" << endl;
  else if (drugi > prvi && ((drugi == 6 && prvi < 5) || drugi == 7))
    cout << "pobedio drugi" << endl;
  else
    cout << "nije zavrseno" << endl;
  return 0;
}
```

9. Navedi broj ispred netačne izjave i obrazlozi zašto izjava nije tačna:

Netačne izjave su:
1, 2, 3, 4, 5, 7, 8, 9

1. ASCII je ravnomeran kod tj. svako karakter se kodira binarnom reči iste dužine.

2. Ispravan poredak reči u poretku rastuće ASCII kolacione sekvence:
007, 111, 13abc, 9, Azbuka, Matematika, abeceda, matematika

3. Leksikografski gledano niska znakova 99 je veća od niske znakova 0099

4. Reč програмирање nije moguće zapisati u osnovnom ASCII.

5. ASCII karakteri koji se koriste za obeležavanje belina imaju kodove 9, 10, 11 13, 32,

7. U osnovnom ASCII kodu, sve uzastopne cifre se ne razlikuju samo u vrednosti jednog bita (npr kodovi za 1,2).

8. Osnovni ASCII kod je sedmobitni kod.

9. U osnovnom ASCII kodu, kontrolni karakteri se ne razlikuju samo u vrednosti jednog bita. (npr kodovi za 7,8,9,10)

Скала оцена

88 - 74 5
73 - 62 4
61 – 45 3
44 – 30 2