Napomena
U svim zadacima, na svim provjerama znanja i vježbama, vrijedi sljedeće: u rješenjima zadataka je upotreba funkcija iz standardne biblioteke dopuštena u svim slučajevima osim kada je to tekstom zadatka eksplicitno zabranjeno.

1. Napisati program koji će s tipkovnice učitati cijeli broj n uz kontrolu da je $3 \leq n \leq 60$. Na slučajan način odabrati n velikih slova (među znakovima A-Z) i ispisati ih na zaslon.

Primjer izvršavanja programa

```
Upisite n > 70↵
Neispravan unos!↵
Upisite n > 0↵
Neispravan unos!↵
Upisite n > 30↵
VYWCQCAKUORJGKZSTDONHWWCUDVVNK
```

2. Napisati program koji će generirati 1,000,000 cijelih brojeva iz intervala [50, 60] i na zaslon ispisati frekvenciju pojavljivanja svakog od brojeva.

Primjer izvršavanja programa

```
50  91096↵
51  91150↵
52  90727↵
53  90539↵
54  91181↵
55  91175↵
56  90356↵
57  90944↵
58  91359↵
59  90826↵
60  90647↵
```

3. Napisati program koji će generirati 5,000,000 realnih brojeva iz intervala [10.0, 15.0] i na zaslon ispisati frekvenciju pojavljivanja brojeva (koristiti širinu razreda 0.5, kao što je prikazano u primjeru).

Primjer izvršavanja programa

```
[10.0 - 10.5) ->  500179↵
[10.5 - 11.0) ->  499302↵
[11.0 - 11.5) ->  499462↵
[11.5 - 12.0) ->  500764↵
[12.0 - 12.5) ->  500210↵
[12.5 - 13.0) ->  500087↵
[13.0 - 13.5) ->  501225↵
[13.5 - 14.0) ->  499376↵
[14.0 - 14.5) ->  499196↵
[14.5 - 15.0] ->  500199↵
```

4. Napisati funkciju `genText` koja na temelju zadanog niza znakova `ulaz` generira sadržaj novog niza znakova zadane duljine `duljinaGen`. Novi niz se generira tako da funkcija na slučajan način odabire znakove iz niza `ulaz` i upisuje ih na mjesto koje je određeno parametrom funkcije `rezultat`, koji pokazuje na niz znakova definiran u pozivajućem programu (za kojeg u pozivajućem programu treba biti rezervirano dovoljno mjesta za upisivanje znakova generiranog niza). Funkcija `genText` mora sama inicijalizirati generator slučajnih brojeva (generator se smije inicijalizirati samo pri prvom pozivu funkcije).

   U glavnom programu (funkciji `main`) učitati niz znakova koji zajedno s eventualno učitanom oznakom novog retka sigurno neće biti dulji od 60 znakova, pomoću funkcije `izbaciNR` (poznatom iz prethodnih vježbi) izbaciti iz niza eventualno učitanu oznaku novog retka, učitati željenu duljinu generiranog novog niza (ne veću od 100, ali nije potrebno kontrolirati), tri puta uzastopno pozvati funkciju `genText` i nakon svakog poziva generirani niz ispisati na zaslon.

   Primjer izvršavanja programa

```
Upisite niz > Birati razne znakove 123632!?.iz OVOG NIZA, stvoriti novi↵
Upisite duljinu generiranog niza > 80↵
eiZer.nai1 tst!?voZiZikir!6riso?2zrzrte?k,iairttnt6 ne?B1izr↵
ioIii? 121n2Zis aVZk1i2IAnNe at 1 k12snzraktIkOiv str.r6vno2↵
ari!Oniz?OoAzOVaozOti?rzOZrov2?1Oni3irZari1iaii!e,3tta,rvBv ↵
```

5. Napisati funkciju `gadjajPolje` koja zadano dvodimenzijsko cjelobrojno polje od `m` redaka i `n` stupaca "gađa zadanim brojem hitaca". Element polja u kojeg pojedini hitac pogađa odabire se na slučajan način (funkcija na slučajan način odabere redak elementa, a zatim na slučajan način odabere stupac elementa). Vrijednost elementa polja koji je "pogođen" funkcija uvećava za jedan. Za inicijalizaciju generatora zadužen je pozivajući program (tj. funkcija računa da je generator pseudoslučajnih brojeva već inicijaliziran). Prije nego počne "gađati" polje, funkcija `gadjajPolje` mora elemente polja inicijalizirati na vrijednost 0.

   U glavnom programu (funkciji `main`) na slučajan način odabrati broj redaka (cijeli broj između 5 i 10) i broj stupaca (cijeli broj između 10 i 20), učitati broj "hitaca kojima treba pogoditi polje", definirati odgovarajuće dvodimenzijsko polje, pozvati funkciju `gadjajPolje` i ispisati rezultat u skladu s primjerom.

Primjer izvršavanja programa

```
Upisite broj hitaca > 1000↵
   1   9   4   7   9   9   8   5  10  12  10   4   6   9   8   9↵
   4  12   8   3   7   7   7  17   8  12   7   9  11   8   6   6↵
  11  11  10  12   5  10   9   8   8  13   9   8   7   9   5   7↵
   9  17   4  11   7  10   5   4   7   9   5  13   9   4  10  11↵
   6   5   7  10  12   5   9   8  11   4   5   8   9   7   7   6↵
  12   3   7   7  11   5   4   8   4   7   5   5   8   7   5  11↵
   6   4   4   9   6   8   7   3  13   2   8  11   7   7  10  11↵
   6   9   6   9   7   6   8  12  10   5   9   6  12   7   8   7↵
```

6. Napisati funkciju `inicijalizirajPolje` koja upisuje znak `'.'` (znak točka, ASCII 46) u sve članove zadane kvadratne matrice mat (članovi su tipa `char`) zadanog reda `n`.

   Napisati funkciju `ispisiPolje` koja u obliku tablice na zaslon ispisuje članove zadane kvadratne matrice mat (članovi su tipa `char`), zadanog reda `n`. Znakove ispisivati prema konverzijskoj specifikaciji `%2c`.

   Napisati funkciju `crtajPutanju` koja u zadanoj kvadratnoj matrice mat (članovi su tipa `char`), zadanog reda `n`, "crta" putanju točke koja se iz gornjeg lijevog kuta matrice nasumično kreće prema dolje ili desno. Putanja se iscrtava znakom koji se funkciji zadaje kao još jedan od parametara. Putanja točke uvijek započinje na indeksu `[0][0]`, a zatim se točka s po 50% vjerojatnosti pomiče ili prema dolje (za jedan redak) ili prema desno (za jedan stupac). Putanja završava u trenutku kada točka dosegne ili zadnji redak ili zadnji stupac matrice. Funkcija može računati na to da je inicijalizacija generatora pseudoslučajnih brojeva obavljena u pozivajućem programu.

   Funkcije `inicijalizirajPolje`, `ispisiPolje` i `crtajPutanju` smjestiti u modul `putanja.c` (uz dodatak datoteke zaglavlja `putanja.h`).

   U glavnom programu (`putanjaGlavni.c`) definirati kvadratnu matricu reda 10 i "napuniti ga točkama". Zatim iscrtati jednu putanju točke, uz korištenje malog slova `'o'` za crtanje traga točke. Na zaslon ispisati rezultat (pomoću funkcije `ispisiPolje`) i jedan prazan red. Zatim u istom polju iscrtati putanju još jedne točke, pri čemu će putanja biti obilježena malim slovom `'x'`. Na zaslon ispisati rezultat.

   Primjer izvršavanja programa

```
 o . . . . . . . . .↵
 o o . . . . . . . .↵
 . o . . . . . . . .↵
 . o o o . . . . . .↵
 . . . o o . . . . .↵
 . . . . o o . . . .↵
 . . . . . o . . . .↵
 . . . . . o . . . .↵
 . . . . . o . . . .↵
 . . . . . o . . . .↵
↵
 x x x x . . . . . .↵
 o o . x . . . . . .↵
 . o . x . . . . . .↵
 . o o x . . . . . .↵
 . . . x x x x . . .↵
 . . . . o o x x . .↵
 . . . . . o . x x .↵
 . . . . . o . . x x↵
 . . . . . o . . . .↵
 . . . . . o . . . .↵
```

7. Načiniti "bubanj za izvlačenje Loto brojeva 7 od 39". U datoteci `loto.c` napisati definicije funkcija čiji su prototipovi prikazani u nastavku:

```
int dajSljedecuKuglicu(void);
void resetirajBubanj(void);
```

Funkcija `dajSljedecuKuglicu` kod svakog poziva treba vratiti sljedeći slučajno odabrani cijeli broj iz intervala [1, 39], s time da se brojevi koji su dobiveni u prethodnim pozivima funkcije ne smiju ponovo pojaviti, tj. tijekom izvlačenja sedam brojeva ne smije iz bubnja dva puta biti izvučena kuglica s istim brojem. Nakon sedmog uzastopnog poziva funkcija više ne vraća brojeve kuglica nego cijeli broj -1. Npr.

1. poziv `dajSljedecuKuglicu()` → 25
2. poziv `dajSljedecuKuglicu()` → 11
3. poziv `dajSljedecuKuglicu()` → 3
4. poziv `dajSljedecuKuglicu()` → 9
5. poziv `dajSljedecuKuglicu()` → 34
6. poziv `dajSljedecuKuglicu()` → 1
7. poziv `dajSljedecuKuglicu()` → 17
8. poziv `dajSljedecuKuglicu()` → -1
9. poziv `dajSljedecuKuglicu()` → -1

itd.

Napomena: za inicijalizaciju generatora pseudoslučajnih brojeva zadužena je funkcija `dajSljedecuKuglicu`. To znači da pozivajući program (u ovom slučaju će to biti funkcija `main`) ne treba inicijalizirati generator pseudoslučajnih brojeva.

Funkcija `resetirajBubanj` vraća (*resetira*) "bubanj" u početno stanje. Nakon poziva funkcije `resetirajBubanj`, uzastopnim pozivima funkcije `dajSljedecuKuglicu` ponovo će se početi dobivati brojevi kuglica, sve do sedmog uzastopnog poziva, nakon čega funkcija `dajSljedecuKuglicu` opet počinje vraćati cijeli broj -1.

U datoteci `lotoGlavni.c` napisati glavni program koji će obaviti 10 uzastopnih izvlačenja Loto 7 od 39 i rezultate ispisati na zaslon u skladu s primjerom.

Primjer izvršavanja programa.

```
 1. izvlacenje:  13 23 16 34 38  2 12↵
 2. izvlacenje:   1 26 23 17 39 30 14↵
 3. izvlacenje:   9 30 33 22 15 38  2↵
 4. izvlacenje:   1 33  8 12 18 17 13↵
 5. izvlacenje:  18  3 38 10 12  5 17↵
 6. izvlacenje:  31 17 18 28 23 35 38↵
 7. izvlacenje:  27 19 17 39 11  3 10↵
 8. izvlacenje:  31 13 22 32  3 39  9↵
 9. izvlacenje:  19 18 16 15  3 32 24↵
10. izvlacenje:  23 16 32  9  1  6 29↵
```

8. Napisati jednostavni program kojim će se pokazati da korištenje funkcije fabsf nad argumentom za kojeg je zapravo trebalo koristiti funkciju fabs, može dovesti do gubitka preciznosti.

Rješenja:

1.
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void) {
   int n;
   do {
      printf("Upisite n > ");
      scanf("%d", &n);
      if (n < 3 || n > 60) {
         printf("Neispravan unos!\n");
      }
   } while (n < 3 || n > 60);

   srand((unsigned)time(NULL));

   int slucajni;
   char ascii;
   for (int i = 0; i < n; ++i) {
      slucajni = rand();
      ascii = slucajni % ('Z' - 'A' + 1) + 'A';
  // ili: ascii = (double)slucajni / (RAND_MAX + 1U) * ('Z' - 'A' + 1) + 'A';
      printf("%c", ascii);
   }

   return 0;
}
```

2.
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define DG 50
#define GG 60
#define BROJ_GEN 1000000

int main(void) {
   srand ((unsigned)time(NULL));

   int slucajni;
   int brojac[GG - DG + 1] = {0};
   for (int i = 0; i < BROJ_GEN; ++i) {
      slucajni = (double)rand() / (RAND_MAX + 1U) * (GG - DG + 1) + DG;
      // ili: slucajni = rand() % (GG - DG + 1) + DG;
      ++brojac[slucajni - DG];
   }
   for (int i = DG; i <= GG; ++i) {
      printf("%3d %6d\n", i, brojac[i - DG]);
   }
   return 0;
}
```

3.
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define DG 10.0f
#define GG 15.0f
#define BROJ_RAZREDA 10
#define BROJ_GEN 5000000

int main(void) {
   srand ((unsigned)time(NULL));

   float slucajni;
   int index, brojac[BROJ_RAZREDA] = {0};
   for (int i = 0; i < BROJ_GEN; ++i) {
      slucajni = (float)rand() / RAND_MAX  * (GG - DG) + DG;
      index = slucajni == GG ? BROJ_RAZREDA - 1 :
                               (int)(slucajni * 2) - (int)DG * 2;
      ++brojac[index];
   }
   for (int i = 0; i < BROJ_RAZREDA; ++i) {
      printf("[%4.1f - %4.1f%c -> %7d\n",
              DG + i / 2.f,
              DG + (i + 1) / 2.f,
              i == BROJ_RAZREDA - 1 ? ']' : ')',
              brojac[i]);
   }

   return 0;
}
```

4.
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>

#define MAXNIZ_ULAZ 60
#define MAXNIZ_GEN 100

void izbaciNR(char *niz) {
   while (*niz != '\0') {
      if (*niz == '\n' && *(niz + 1) == '\0') {
         *niz = '\0';
      }
      ++niz;
   }
   return;
}

void genText(char *ulaz, int duljinaGen, char *rezultat) {
   static bool inicijaliziran = 0;
   if (!inicijaliziran) {
      srand((unsigned int)time(NULL));
      inicijaliziran = 1;
   }
   int duljinaUlaz = 0;
   while (*(ulaz + duljinaUlaz) != '\0') {
      ++duljinaUlaz;
   }
   for (int i = 0; i < duljinaGen; ++i) {
      *rezultat = *(ulaz + rand() % duljinaUlaz);
      ++rezultat;
   }
   *rezultat = '\0';

   return;
}

int main(void) {
   char ulazni[MAXNIZ_ULAZ + 1];
   char generirani[MAXNIZ_GEN + 1];
   int duljinaGen;

   printf("Upisite niz > ");
   fgets(ulazni, MAXNIZ_ULAZ + 1, stdin);
   izbaciNR(ulazni);

   printf("Upisite duljinu generiranog niza > ");
   scanf("%d", &duljinaGen);

   for (int i = 0; i < 3; ++i) {
      genText(ulazni, duljinaGen, generirani);
      printf("%s\n", generirani);
   }

   return 0;
}
```

5.
```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define DG_M 5
#define GG_M 10
#define DG_N 10
#define GG_N 20

void gadjajPolje(int *polje, int m, int n, int brojHitaca) {
   for (int i = 0; i < m; ++i)
      for (int j = 0; j < n; ++j)
         *(polje + n * i + j) = 0;

   for (int i = 0; i < brojHitaca; ++i) {
      int slucajniRedak = rand() % m;
      int slucajniStupac = rand() % n;
      ++*(polje + n * slucajniRedak + slucajniStupac);
   }
}

int main(void) {
   int m, n;

   srand((unsigned)time(NULL));
   m = rand() % (GG_M - DG_M + 1) + DG_M;
   n = rand() % (GG_N - DG_N + 1) + DG_N;
   int polje[m][n];

   int brojHitaca;
   printf("Upisite broj hitaca > ");
   scanf("%d", &brojHitaca);

   gadjajPolje(&polje[0][0], m, n, brojHitaca);

   for (int i = 0; i < m; ++i) {
      for (int j = 0; j < n; ++j) {
         printf("%4d", polje[i][j]);
      }
      printf("\n");
   }

   return 0;
}
```

6. **putanja.c**

```
#include <stdio.h>
#include <stdlib.h>
#include "putanja.h"

void inicijalizirajPolje(char *polje, int n) {
   for (int i = 0; i < n; ++i) {
      for (int j = 0; j < n; ++j) {
         *(polje + n * i + j) = '.';
      }
   }
}

void crtajPutanju(char *polje, int n, char znak) {
   int redak = 0, stupac = 0;
   *(polje + n * redak + stupac) = znak;

   int smjer;
   while (redak < n - 1 && stupac < n - 1) {
      smjer = rand() % 2; // ako je 0-idi dolje, ako je 1-idi desno
      if (smjer == 0) {
         ++redak;
      } else {
         ++stupac;
      }
      *(polje + n * redak + stupac) = znak;
   }
}

void ispisiPolje(char *polje, int n) {
   for (int i = 0; i < n; ++i) {
      for (int j = 0; j < n; ++j) {
         printf("%2c", *(polje + n * i + j));
      }
      printf("\n");
   }
}
```

putanja.h

```
void inicijalizirajPolje(char *polje, int n);
void crtajPutanju(char *polje, int n, char znak);
void ispisiPolje(char *polje, int n);
```

putanjaGlavni.c

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "putanja.h"

#define N 10

int main(void) {
   char polje[N][N];

   inicijalizirajPolje(&polje[0][0], N);

   srand((unsigned)time(NULL));

   crtajPutanju(&polje[0][0], N, 'o');
   ispisiPolje(&polje[0][0], N);

   printf("\n");
   crtajPutanju(&polje[0][0], N, 'x');
   ispisiPolje(&polje[0][0], N);

   return 0;
}
```

7. **loto.c**

```
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include "loto.h"

#define MAX_KUGLICA 7
#define KUGL_U_BUBNJU 39

static int doSadaIzvucenoBrojeva = 0;

int dajSljedecuKuglicu(void) {
   static int doSadaIzvuceniBrojevi[MAX_KUGLICA];

   static bool inicijaliziranGenPseudo = 0;
   if (!inicijaliziranGenPseudo) {
      srand((unsigned int)time(NULL));
      inicijaliziranGenPseudo = 1;
   }

   int izvuceniBroj;
   if (doSadaIzvucenoBrojeva < MAX_KUGLICA) {
      bool jestDuplikat;
      do {
         izvuceniBroj = (double)rand() / (RAND_MAX + 1U) * (KUGL_U_BUBNJU) + 1;
         jestDuplikat = 0;
         for (int i = 0; i < doSadaIzvucenoBrojeva; ++i) {
            if (izvuceniBroj == doSadaIzvuceniBrojevi[i]) {
               jestDuplikat = 1;
            }
         }
      } while (jestDuplikat);
      doSadaIzvuceniBrojevi[doSadaIzvucenoBrojeva] = izvuceniBroj;
      ++doSadaIzvucenoBrojeva;
   } else {
      izvuceniBroj = -1;
   }

   return izvuceniBroj;
}

void resetirajBubanj(void) {
   doSadaIzvucenoBrojeva = 0;

   return;
}
```

loto.h

```
int dajSljedecuKuglicu(void);
void resetirajBubanj(void);
```

lotoGlavni.c

```
#include <stdio.h>
#include "loto.h"

#define BROJ_IZVLACENJA 10

int main(void) {
   for (int i = 1; i <= BROJ_IZVLACENJA; ++i) {
      printf("%2d. izvlacenje: ", i);
      resetirajBubanj();
      for (int j = 0; j < 7; ++j) {
         printf("%3d", dajSljedecuKuglicu());
      }
      printf("\n");
   }

   return 0;
}
```

8.
```
#include <stdio.h>
#include <math.h>

int main(void) {
   double x = -12345.1234567;
   double rez1 = fabsf(x);
   double rez2 = fabs(x);
   printf("%14.7lf\n", rez1);
   printf("%14.7lf\n", rez2);

   return 0;
}
```