Pogađanje brojeva

- Napišite program u kojem pogađate generiran slučajan broj. Upisivanjem broja dobivate obavijest da li je on veći ili manji od traženog broja.
  Funkcija rand() vraća slučajan broj između 0 i RAND_MAX. (Obično je RAND_MAX jednak 32767).
  Ako želimo dobiti slučajan prirodan broj između m i n (m < n) uključujući i granice m i n, tada treba napraviti ovo:
  int x = m + (n - m + 1) * ((float)rand() / (RAND_MAX + 1));
  to su pseudoslučajni brojevi,
- potreban različit začetak za različite nizove slučajnih brojeva, što se radi s:
  srand(time(NULL));

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main(){
	int br;
	int pogodak = 0;
	srand(time(NULL)*10);
	int x =  1 + 100 * ((float)rand() / (RAND_MAX + 1));
	printf("Zamislio sam broj izmedju 1 i 100. Pogodi ga!\n");
	int brpok = 0;
	while(pogodak == 0)	{
		printf("Unesi broj: ");
		scanf("%d", &br);
		brpok++;
		if(br == x) {
			printf("Bravo! Pogodio si u pokusaju br. %d\n", brpok);
			pogodak = 1;
		} else	{
			if (x > br)
				printf("Trazeni broj je veci!\n");
			else
				printf("Trazeni broj je manji!\n!!");
		}	}
	  system("PAUSE");
	return 0;
}
```

Empirijska provjera najčešćeg ishoda slučajnog pokusa bacanja dviju kocaka

- Lako se može izračunati (elementarna teorija vjerojatnosti) da je pri bacanju dviju kocaka najvjerojatnije da se kao zbroj točkica na kockama pojavi zbroj 7. Lako je vidjeti da vjerojatnost tog događaja iznosi p = 6/36 = 0.166666...
  Napisati program koji će i empirijski provjeriti ovaj teorijski rezultat. Program treba veliki broj puta (npr. milijun ili više puta) simulirati bacanje dviju kocaka, na prikladan način spremati ishode tih bacanja i na kraju prebrojiti i ispisati kojih ishoda je bilo najviše, te izračunati i ispisati relativnu frekvenciju tog (najčešćeg) ishoda.
  Dakle, moguće situacije su:

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define BR_PONAVLJANJA 1000000
int IndeksNajveceg(int a[], int length);
int main(){
	int frekv[13] = { 0 };
	srand(time(NULL));
	printf("Bacam kocke...\n");
	for(int i = 0; i < BR_PONAVLJANJA; i++){
		int kocka1 = 1 + 6 * ( (float)rand() / (RAND_MAX + 1) );
		int kocka2 = 1 + 6 * ( (float)rand() / (RAND_MAX + 1) );
		frekv[kocka1 + kocka2]++;
	}
	int inajv = IndeksNajveceg(frekv, 13);
	float relfrekv = (float)frekv[inajv] / BR_PONAVLJANJA;
	printf("Najcesci zbroj pri bacanju dviju kocki je %d\n", inajv);
	printf("Rel. frekvencija pojave zbroja %d je %f\n", inajv, relfrekv);
	system("PAUSE"); return 0;
}
int IndeksNajveceg(int a[], int length){
	int indmax = 0;
	for(int i = 1; i < length; i++)
		if (a[i] > a[indmax]) indmax = i;
	return indmax;
}
```

Rad s nizom znakova

- Napravite niz duljine do 80 znakova koje unesete, upišete znak koji tražite u niz i napišite program koja pretražuje niz znakova i javlja poziciju na kojoj se traženi znak nalazi, ukoliko se traženi znak ne nalazi u nizu, obavještava Vas o tome

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define L 80
int traziznak(char *niz, char *znak, int nizl);

int main(){
      char *tekst;
      char *ctrazeni;
      int poz=0;
      int nizl;
      if ((tekst = (char *)malloc(L+1)) == NULL)
            {printf("Greska u alociranju memorije !");
          exit (1); }
if ((ctrazeni = (char *)malloc(2)) == NULL)
            {printf("Greska u alociranju memorije !");
          exit (1); }
```

```
    printf("Unesi do %d znaka:\n",L);
    gets(tekst);
    printf("Unesi znak koji trazis:\n");
    gets(ctrazeni);
    nizl=strlen(tekst);
    printf("Uneseni niz ima %d znakova\n",nizl);

    poz = traziznak(tekst,ctrazeni,nizl);
    if (!poz)
            printf("Znak %s se ne nalazi u nizu %s\n",ctrazeni,tekst);
            else
    printf ("Znak %s je na mjestu %d u nizu %s\n",ctrazeni,poz,tekst);
    system("PAUSE");
    return 0;
}

int traziznak(char *niz, char *znak, int nizl){
    int ind;
    for (ind=0; ind <= nizl; ind++)
            if (niz[ind] == *znak) return ind+1;
    return 0;
}
```

Dvodimenzionalna i višedimenzionalna polja kao argumenti funkcije

- Polje u funkciji treba prihvatiti kao jednodimenzionalno polje ili pokazivač.
- Primjer:

```
int polje[3][4] = { { 1, 2, 3, 4},
                    { 5, 6, 7, 8},
                    { 9, 10, 11, 12} };
```

u memoriji računala spremljeno je kao

tj. jednako kao jednodimenzionalno polje od 12 elemenata.

- Razlikuju se fizičke dimenzije polja od logičkih, koje mogu biti i manje. Za pozicioniranje je potrebno znati fizički broj stupaca `(MAX_broj_stupaca)`.
- Za dohvat elementa iz *i*-tog retka treba preskočiti `i-1` punih redaka.
- Kako prvi redak ima indeks 0, drugi 1, treći 2 itd., za dohvat elemenata retka s indeksom *i* treba preskočiti `i*MAX_broj_stupaca` članova polja.
- Općenito:

```
dvodim_polje[i][j] ≡ jednodim_polje[i * MAX_broj_stupaca + j]
```

Zbrajanje i množenje matrica

- Napišite program koji će rezervirati u memoriji mjesto za tri dvodimenzionalna polja (matrice) A, B i C dimenzija maxstu. Polja A i B popuniti slučajno generiranim cijelim brojevima u rasponu od 0 do maxcl. Napisati funkciju koja zbroji matrice A i B i rezultat pospremi u polje C. Napisati funkciju koja umnožak matrica A*B pospremi u polje C. Ispisati sva polja.

- Suma: $c_{ij} = a_{ij} + b_{ij}$

- Produkt: $$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

```
#include <stdlib.h>
#include <time.h>
#include <stdio.h>
#define maxstu 4
#define maxcl 5

void prikazi(int *M){
int i,j;

for (i=0;i<maxstu;i++)  {
   for (j=0;j<maxstu;j++)
       printf("%d ",*(M+i*maxstu+j));
   printf("\n");
  }
printf("\nPritisnite Enter za nastavak!\n");
scanf("%*c");
}

void zbroji(int *A,int *B,int *C){
int i,j;

 for (i=0;i<maxstu;i++)
    for (j=0;j<maxstu;j++)
      *(C+i*maxstu+j)=*(A+i*maxstu+j)+*(B+i*maxstu+j);
}
```

```
void mnozi(int *A,int *B,int *C){
int i,k,j;

 for (i=0;i<maxstu;i++)
    for (j=0;j<maxstu;j++)
       for (k=0;k<maxstu;k++)
     *(C+i*maxstu+j)+=*(A+i*maxstu+k) * *(B+k*maxstu+j);
}

int main(){
int i,j;
int A[maxstu][maxstu];
int B[maxstu][maxstu];
int C[maxstu][maxstu]={0};

srand(time(NULL));
for (i=0;i<maxstu;i++)
   for (j=0;j<maxstu;j++)   {
      srand(time(NULL) * (float) rand() / (RAND_MAX+1));
      A[i][j]=maxcl * ((float) rand() / (RAND_MAX+1));
      B[i][j]=maxcl * ((float) rand() / (RAND_MAX+1));
   }
```

```
printf("\n------------------A------------------\n\n");
prikazi(&A[0][0]);
printf("\n------------------B------------------\n\n");
prikazi(&B[0][0]);
printf("\n------------------C------------------\n\n");
prikazi(&C[0][0]);

zbroji(&A[0][0],&B[0][0],&C[0][0]);
printf("\n----------------C=A+B------------------\n\n");
prikazi(&C[0][0]);

for (i=0;i<maxstu;i++)
  for (j=0;j<maxstu;j++)
      C[i][j]=0;

mnozi(&A[0][0],&B[0][0],&C[0][0]);
printf("\n----------------C=A*B------------------\n\n");
prikazi(&C[0][0]);

return 0;
}
```

Zadaci za zadaću

- Napisati program koji učita 10 nizova znakova duljine do 20 znakova. Svi unešeni znakovi su samo alfabetski standardni ASCII znakovi (mala i velika slova). Sortirati te nizove znakova po abecedi i ispisati ih. Uputa: prebaciti sve unešene znakove u velika ili mala slova prije sortiranja i tako ih sortirati i ispisati.

- Napisati program koji će upotrebom velikog broja generiranih parova slučajnih brojeva izračunati površinu kruga radijusa r s centrom u ishodištu koordinatnog sustava. Dobiveni rezultat usporediti s izračunatom pravom vrijednošću površine kruga.