11. Dodatne vježbe

1. S tipkovnice učitati 10 cijelih brojeva u polje, redoslijed članova **zamijeniti unutar** polja. Nakon što se redoslijed članova u polju promijeni, ispisati novi sadržaj polja. Npr. ako su u polje učitani brojevi
   `1 3 5 7 9 11 13 15 17 19`
   njihov redoslijed u polju treba promijeniti tako da su članovi polja sljedeći:
   `19 17 15 13 11 9 7 5 3 1`

2. Učitati pozitivni cijeli broj. Uzastopnim dijeljenjem sa 16 odrediti znamenke ekvivalentnog heksadekadskog broja. Ispisati ekvivalentni heksadekadski broj, ali tako da su znamenke heksadekadskog broja ispisane ispravnim redoslijedom. **Naputak:** znamenke ubacivati u polje. Ispisati članove polja obrnutim redoslijedom. Koliko članova mora imati polje pri definiciji da bi se mogla obaviti pretvorba bilo kojeg pozitivnog cijelog broja tipa *signed int* (pretpostavlja se da *int* koristi 4 okteta).

3. Cjelobrojno polje je potrebno napuniti s prvih 40 članova Fibonaccijevog niza. **Naputak:** za prva dva člana polja ubaciti 1 i 1. Svaki sljedeći član izračunati na temelju prethodna dva. Uočite da se prethodna dva člana uvijek nalaze u polju, pa nije potrebno koristiti varijable f0 i f1 kao u primjeru s predavanja. Kakvu petlju ovdje treba koristiti?

4. Cjelobrojno polje je potrebno napuniti s 500 prostih brojeva. Koristiti činjenicu da je neki broj x prost ukoliko nije djeljiv s niti jednim prostim brojem koji je manji od x (osim jedinice). Nakon punjenja polja ispisati sadržaj polja. **Naputak:** u polje kao prva dva člana ubaciti 1 i 2. Za svaki sljedeći neparni broj ("broj kandidat") ispitivati je li djeljiv s bilo kojim prostim brojem koji se do tada već nalazi u polju prostih brojeva (pri tome uvijek preskočiti prvi i drugi član, tj. 1 i 2). Vanjskom petljom povećava se "broj kandidat", a unutarnjom petljom se testira djeljivost "broja-kandidata" sa svim prethodno upisanim prostim brojevima osim 1. i 2. člana. Kakve se petlje ovdje moraju koristiti?

   Ovo jest malo teži zadatak, ali vrijedi ga pokušati riješiti ili barem analizirati rješenje.

5. Pomoću funkcije `gets` učitajte s tipkovnice niz znakova koji sigurno nije dulji od 30 znakova. Izračunajte i ispišite koliko u učitanom nizu ima samoglasnika, koliko suglasnika, a koliko ostalih znakova.
   Primjer:
   ako korisnik upiše sljedeći niz: **`printf("%s", "Ana");`**`<Enter>` na zaslon treba ispisati:
   `Samoglasnika: 3`
   `Suglasnika: 7`
   `Ostalih: 10`

6. S tipkovnice, u obliku dekadskih brojeva, učitati najviše 8 ASCII vrijednosti znakova. Učitavanje prekinuti i onda kada korisnik unese ASCII vrijednost koja ne predstavlja veliko slovo abecede. Učitane vrijednosti pohraniti u polje znakova koje se koristi kao niz znakova. Na kraju, ispisati dobiveni niz znakova pomoću funkcije printf i formata %s.
   Primjeri:
   ako korisnik upiše sljedeći niz ASCII vrijednosti:
   `80 69 82 85 64 80 69``<Enter>` na zaslon treba ispisati **`PERU`**
   ako korisnik upiše sljedeći niz ASCII vrijednosti:
   `65 82 71 69 78 84 73 78 65 66``<Enter>` na zaslon treba ispisati **`ARGENTIN`**

7. S tipkovnice učitati cijeli broj **`n`** koji mora biti iz intervala [3, 10]. Učitavanje broja **`n`** ponavljati dok god ne bude ispravno upisan. Nakon toga s tipkovnice učitati **`n`** cijelih brojeva i pohraniti ih u polje **`ulaz`**. U polje **`parni`** prepisati sve parne vrijednosti, a u polje **`neparni`** prepisati sve neparne vrijednosti iz polja **`ulaz`**. Na kraju ispisati članove polja **`ulaz`**, **`parni`** i **`neparni`** (svako polje u svom retku, s članovima međusobno odvojenim prazninom).

Rješenja svih zadataka provjeriti prevođenjem i testiranjem vlastitih programa!

Rješenja: NE GLEDATI prije nego sami pokušate riješiti zadatke

Rješenje 1. zadatka

```
#include <stdio.h>

#define MAXCLAN 10
int main() {
   int i;
   int broj[MAXCLAN];
   int pom;

   for (i = 0; i < MAXCLAN; i++)
      scanf("%d", &broj[i]);

   for (i = 0; i < MAXCLAN/2; i++) {
      pom = broj[i];
      broj[i] = broj[MAXCLAN-1-i];
      broj[MAXCLAN-1-i] = pom;
   }

   for (i = 0; i < MAXCLAN; i++)
      printf("%d\n", broj[i]);

   return 0;
}
```

Rješenje 2. zadatka

```
#include <stdio.h>

#define MAXZNAM 8
int main() {
   int i, broj;
   char znamenke[MAXZNAM];
   int brojZnam = 0;
   char znamenka;

   scanf("%d", &broj);

   while (broj > 0) {
      znamenka = broj % 16;
      /* sad pretvoriti broj 0-15 u znak '0'-'9' ili 'A'-'F' i ubaciti ga u polje*/
      if (znamenka <= 9)
         znamenke[brojZnam] = '0' + znamenka;
      else
         znamenke[brojZnam] = 'A' + znamenka - 10;

      broj /= 16;
      brojZnam++;
   }

   for (i = brojZnam-1; i >= 0; i--)
      printf("%c", znamenke[i]);

   printf("\n");
   return 0;
}
```

Pretpostavilo se da će SIGURNO biti učitan pozitivan cijeli broj (veći od nule). Varijabla brojZnam sadrži podatak o tome koliko heksadekadski broj ima znamenaka. Heksadekadski broj pohranjen u 4 bajta ima najviše 8 znamenaka.

S obzirom da heksadekadske znamenke ne mogu biti prikazane znamenkama 0-9, koristi se polje znakova. Postupkom dijeljenja prvo se dobiva znamenka u rasponu 0-15, a zatim se ona transformira u ASCII vrijednost odgovarajućeg znaka ('0' ... '9' i 'A' ... 'F'). Ovakva transformacija ne bi bila potrebna da se broj pretvarao u broj s bazom ≤ 10. Tada bi se znamenke mogle pohraniti kao "obični" brojevi, a kod ispisa svake znamenke bi se koristio format %d. Možete probati npr. s oktalnim brojevnim sustavom.

Problem se mogao riješiti i tako da se znamenke heksadekadskog broja u polje ubacuju "odostraga". Tada bi se članovi polja mogli ispisati redom ("sprijeda"), s time da bi trebalo paziti od kojeg člana polja se počinje s ispisom. Alternativno, svi članovi polja bi se mogli na početku napuniti znakom '0', a onda heksadekadski broj ispisivati s vodećim nulama. Pokušajte sami.

Rješenje 3. zadatka

```
#include <stdio.h>

#define MAXCLAN 40
int main() {
   int i;
   int fbroj[MAXCLAN];

   fbroj[0] = fbroj[1] = 1;

   for (i = 2; i < MAXCLAN; i++)
      fbroj[i] = fbroj[i-1] + fbroj[i-2];

   for (i = 0; i < MAXCLAN; i++)
      printf("%d\n", fbroj[i]);
   return 0;
}
```

Rješenje 4. zadatka

```
#include <stdio.h>

#define MAX 500
#define TRUE 1
#define FALSE 0

int main() {
   int prosti[MAX];
   int slobodanInd, indProstog, kandidat, jeProst, i;
   prosti[0] = 1;
   prosti[1] = 2;
   slobodanInd = 2;
   kandidat = 3;
   while (slobodanInd < MAX) {
      jeProst = TRUE;
      for (indProstog = 3; indProstog < slobodanInd; indProstog++)
         if( kandidat % prosti[indProstog] == 0 ) {
            jeProst = FALSE;
            break;
         }
      if (jeProst)
         prosti[slobodanInd++] = kandidat;
      kandidat += 2;
   }

   /* kontrolni ispis */
   for (i = 0; i < MAX; i++)
      printf("%d. %d\n", i+1, prosti[i]);

   return 0;
}
```

Varijabla slobodanInd ima vrijednost indeksa prvog slobodnog člana u polju prostih brojeva. Varijabla kandidat sadrži broj kojeg treba testirati da li je prost. Ta varijabla poprima vrijednosti 3, 5, 7, ...

Postupak uvećanja varijable kandidat, njenog testiranja i eventualnog ubacivanja u polje ponavlja se dok god se polje ne popuni s 500 prostih brojeva (tj. dok je slobodanInd < MAX).

Testiranje kandidata se obavlja tako da se ispituje ostatak dijeljenja kandidata sa svakim prostim brojem koji se već nalazi u polju (osim prosti[0] i prosti[1]). Kad se (ako se) naiđe na prosti broj s kojim je kandidat djeljiv, varijabla jeProst se postavlja na FALSE, a petlja se prekida. Ako je varijabla jeProst tijekom obavljanja for petlje ostala TRUE, tada se kandidat ubacuje u polje kao novi prosti broj.

Rješenje 5. zadatka

```
#include <stdio.h>
int main() {
   char niz[30+1]; /* vazno pitanje: zasto + 1? */
   int samog = 0, sug = 0, ostali = 0;
   int i = 0;
   gets(niz);

   while (niz[i] != '\0') {
      if (niz[i]=='A' || niz[i]=='E' || niz[i]=='I' || niz[i]=='O' || niz[i]=='U' ||
          niz[i]=='a' || niz[i]=='e' || niz[i]=='i' || niz[i]=='o' || niz[i]=='u')
         samog++;
      else if (niz[i]>='A' && niz[i]<='Z' || niz[i]>='a' && niz[i]<='z')
         sug++;
      else
         ostali++;
      i++;
   }
   printf("Samoglasnika: %d\n", samog);
   printf("Suglasnika: %d\n", sug);
   printf("Ostalih: %d\n", ostali);
   return 0;
}
```

Rješenje 6. zadatka

```
#include <stdio.h>
#define MAXNIZ 8

int main() {
   char niz[MAXNIZ+1];
   int ascii, i = 0;
   do {
      scanf("%d", &ascii);
      if (ascii >= 'A' && ascii <= 'Z') {
         niz[i] = ascii;
         i++;
      }
   } while (i < MAXNIZ && ascii >= 'A' && ascii <= 'Z');

   /* zasto je sljedeca naredba vazna? */
   niz[i] = '\0';

   printf("%s\n", niz);
   return 0;
}
```

Rješenje 7. zadatka

```
#include <stdio.h>
#define MAXPOLJE 10

int main() {
   int ulaz[MAXPOLJE];
   int parni[MAXPOLJE];
   int neparni[MAXPOLJE];
   int n, i, brojParnih = 0, brojNeparnih = 0;

   do {
      printf("Unesite n: ");
      scanf("%d", &n);
   } while (n < 3 || n > 10);

   for (i = 0; i < n; i++) {
      printf("Unesite %d. clan: ", i+1);
      scanf("%d", &ulaz[i]);
   }

   for (i = 0; i < n; i++)
      if (ulaz[i] % 2 == 0)
         parni[brojParnih++] = ulaz[i];
      else
         neparni[brojNeparnih++] = ulaz[i];

   printf("\nPolje ulaz: ");
   for (i = 0; i < n; i++)
      printf("%d ", ulaz[i]);

   printf("\nPolje parni: ");
   for (i = 0; i < brojParnih; i++)
      printf("%d ", parni[i]);

   printf("\nPolje neparni: ");
   for (i = 0; i < brojNeparnih; i++)
      printf("%d ", neparni[i]);

   printf("\n");

   return 0;
}
```