

Napomene:

- Savjetuje se navedene zadatke riješiti odmah nakon predavanja
- Savjetuje se ne gledati rješenja prije nego što se pokuša samostalno riješiti zadatke

12. Vježbe uz predavanja

1. Pomoću funkcije gets s tipkovnice učitati niz znakova koji sigurno nije dulji od 80 znakova. Za svako malo slovo engleske abecede (znakovi a-z) ispisati koliko se puta pojavilo u učitanoj nizu.

```
slovo 'a' pojavilo se xx puta  
slovo 'b' pojavilo se xx puta  
slovo 'c' pojavilo se xx puta  
...  
slovo 'z' pojavilo se xx puta
```

2. S tipkovnice, redak po redak, učitati članove cjelobrojne matrice dimenzija 4 retka i 3 stupca. Ispisati matricu (u obliku dvodimenzionalne tablice), te aritmetičku sredinu vrijednosti članova matrice.
3. Definirati i inicijalizirati trodimenzionalno cjelobrojno polje s dimenzijama koje sami odaberite (npr. 3, 4, 5). Koristiti inicijalizator s vitičastim zagradama (svaki sloj unutar svojih vitičastih zagrada, svaki redak sloja unutar svojih vitičastih zagrada). Ispišite polje po slojevima, u sljedećem obliku:

```
x x x x x  
x x x x x  
x x x x x  
x x x x x  
  
x x x x x  
x x x x x  
x x x x x  
x x x x x  
  
x x x x x  
x x x x x  
x x x x x  
x x x x x
```

Rješenja svih zadataka provjeriti prevođenjem i testiranjem vlastitih programa!

Rješenja:

Rješenje 1. zadatka - varijanta u kojoj se frekvencije 'pamte' u polju

```
#include <stdio.h>

#define MAXNIZ 80
#define DG 'a'
#define GG 'z'

int main() {
    char niz[MAXNIZ+1], pozicija_u_nizu, c_iz_abecede;
    int brojac[GG-DG+1] = {0};
    gets(niz);

    pozicija_u_nizu = 0;
    while (niz[pozicija_u_nizu] != '\0') {
        brojac[niz[pozicija_u_nizu]-DG]++;
        pozicija_u_nizu++;
    }
    for (c_iz_abecede = DG; c_iz_abecede <= GG; c_iz_abecede++) {
        printf("slovo '%c' pojavilo se %2d puta\n",
            , c_iz_abecede
            , brojac[c_iz_abecede - DG]);
    }

    return 0;
}
```

Rješenje 1. zadatka - varijanta u kojoj se ne koristi polje za pohranu frekvencija

```
#include <stdio.h>

#define MAXNIZ 80
#define DG 'a'
#define GG 'z'

int main() {
    char niz[MAXNIZ+1], pozicija_u_nizu, c_iz_abecede;
    int brojac;
    gets(niz);

    for (c_iz_abecede = DG; c_iz_abecede <= GG; c_iz_abecede++) {
        pozicija_u_nizu = 0;
        brojac = 0;
        while (niz[pozicija_u_nizu] != '\0') {
            if (niz[pozicija_u_nizu] == c_iz_abecede) {
                brojac++;
            }
            pozicija_u_nizu++;
        }
        printf("slovo '%c' pojavilo se %2d puta\n", c_iz_abecede, brojac);
    }
}
```

```
    }  
    return 0;  
}
```

Rješenje 2. zadatka

```
#include <stdio.h>  
  
#define MAXRED 4  
#define MAXSTUP 3  
  
int main() {  
    int i, j, suma = 0;  
    int mat[MAXRED][MAXSTUP];  
    for (i = 0; i < MAXRED; i++) {  
        /* u i-tom retku obavi sljedece */  
        for (j = 0; j < MAXSTUP; j++) {  
            /* u j-tom stupcu i-tog retka obavi sljedece */  
            printf("Upisite clan matrice [%d][%d]->", i, j);  
            scanf("%d", &mat[i][j]);  
            suma+=mat[i][j];  
        }  
    }  
    for (i = 0; i < MAXRED; i++) {  
        for (j = 0; j < MAXSTUP; j++) {  
            printf("%5d", mat[i][j]);  
        }  
        printf("\n");  
    }  
    printf("Ar. sredina je %.2f\n", (float) suma / (MAXRED * MAXSTUP));  
    return 0;  
}
```

Rješenje 3. zadatka

```
#include <stdio.h>

#define MAXSLOJ 3
#define MAXRED 4
#define MAXSTUP 5

int main() {
    int i, j, k;
    int tridim[MAXSLOJ][MAXRED][MAXSTUP] =
        {
            { { 101, 102, 103, 104, 105},
              { 106, 107, 108, 109, 110},
              { 111, 112, 113, 114, 115},
              { 116, 117, 118, 119, 120 }
            },
            { { 201, 202, 203, 204, 205},
              { 206, 207, 208, 209, 210},
              { 211, 212, 213, 214, 215},
              { 216, 217, 218, 219, 220 }
            },
            { { 301, 302, 303, 304, 305},
              { 306, 307, 308, 309, 310},
              { 311, 312, 313, 314, 315},
              { 316, 317, 318, 319, 320 }
            }
        };

    for (i = 0; i < MAXSLOJ; i++) {
        for (j = 0; j < MAXRED; j++) {
            for (k = 0; k < MAXSTUP; k++)
                printf("%d ", tridim[i][j][k]);
            printf("\n");
        }
        printf("\n");
    }

    return 0;
}
```