STRUKTURE PODATAKA I ALGORITMI 1

VEŽBE 6

Nikola Bačanin
Nikola Rnjak

PRIMER 1 – MALLOC()

Napisati program koji unosi niz proizvoljne dimenzije i nalazi najveci element – Verzija 1

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int n, *a, i, max;
    printf("Unesi dimenziju niza : "); scanf("%d", &n);
    a = (int *)malloc(n * sizeof(int));
    if (a == NULL) {
        printf("Greska : Nema dovoljno memorije!\n");
        return 1;
    }
    for (i = 0; i < n; i++) {
        printf("a[%d]=", i); scanf("%d", &a[i]);
    }
    // PRONALAZENJE MAKSIMUMA NIZA
    free(a);
    return 0;
}
```

PRIMER 2 – MALLOC()

Napisati program koji unosi niz proizvoljne dimenzije i nalazi najveci element – Verzija 2

```
#include <stdio.h>
#include <stdlib.h>
int *KreirajNiz(int n) {
    return (int *)malloc(n * sizeof(int));
}
int main() {
    int n, *a, i, max;
    printf("Unesi dimenziju niza : "); scanf("%d", &n);
    a = KreirajNiz(n);
    if (a == NULL) {
        printf("Greska : Nema dovoljno memorije!\n");
        return 1;
    }
    // Nadalje a koristimo kao obican niz
    free(a);
    return 0;
}
```

PRIMER 3 – CALLOC()

Program demonstrira funkciju calloc() - funkcija inicijalizuje sadrzaj memorije na 0.

```
#include <stdio.h>
#include <stdlib.h>
#define BR_ELEM 10
main()
{
   int *m, *c, i;
   m = malloc(BR_ELEM*sizeof(int));
   c = calloc(BR_ELEM, sizeof(int));
   for (i = 0; i<BR_ELEM; i++)
      printf("m[%d] = %d\n", i, m[i]);
   for (i = 0; i<BR_ELEM; i++)
      printf("c[%d] = %d\n", i, c[i]);
   free(m);
   free(c);
}
```

PRIMER 4 – REALLOC()

Program demonstrira niz kome se velicina tokom rada povecava.

```
#include <stdio.h>
#include <stdlib.h>
#define KORAK 10
int main() {
    int *a = NULL;
    int duzina = 0, alocirano = 0;
    int n, i;
    do {
        printf("Unesi ceo broj (-1 za kraj): ");
        scanf("%d", &n);
        if (duzina == alocirano) {
            alocirano = alocirano + KORAK;
            a = realloc(a, alocirano * sizeof(int));
        }
        a[duzina++] = n;
    } while (n != -1);
```

PRIMER 4 – REALLOC()

Program demonstrira niz kome se velicina tokom rada povecava. Nastavak...

```
    printf("Uneto je %d brojeva. Alocirano je ukupno "
        "%ld bajtova\n", duzina, alocirano * sizeof(int));
    printf("Brojevi su : ");
    for (i = 0; i < duzina; i++)
        printf("%d ", a[i]);
    free(a);
    return 0;
}
```

VRACANJE NIZA IZ FUNKCIJE

```
#include <stdio.h>
#include <stdlib.h>
int* KreirajNiz(int n) {
   return (int*)malloc(n * sizeof(int));
}
int* UnosNiza(int n) {
   int *x, i;
   x = KreirajNiz(n);
   for(i = 0; i < n; i++) scanf("%d", &x[i]);
   return x;
}
int main() {
   int i, n, *a;
   printf("Unesi dimenziju niza "); scanf("%d", &n);
   a = UnosNiza(n);
   for(i = 0; i < n; i++) printf("%5d", a[i]);
   printf("\n");
   free(a);
}
```

STATIČKA ALOKACIJA PROSTORA ZA MATRICU

```c
#include <stdio.h>
int main() {
    int a[3][3], i, j;
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++) {
            printf("a[%d][%d] = ", i, j);
            scanf("%d", &a[i][j]);
            /* scanf("%d", a[i] + j); */
            /* scanf("%d", *(a + i) + j); */
        }
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++) {
            printf("%5d", a[i][j]);
            /* printf("%5d", *(a[i] + j)); */
            /* printf("%5d", *(*(a + i) + j)); */
        }
        printf("\n");
    }
}
```

```c
#include <stdio.h>
int main() {
    int i, j;
    int a[3][3] = {
        {  0,  1,  2 },
        { 10, 11, 12 },
        { 20, 21, 22 }};

    int (*pa)[3]; // int *pa[3]; -> pogresno!!!
    pa = a;
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++)
            printf("%5d", pa[i][j]);
        printf("\n");
    }
    printf("%p\n", pa);
    printf("%p\n", pa[0]);
    printf("%p\n", pa[1]);
}
```

```c
#include <stdio.h>
int main() {
    int i, j;
    int a[3][3] = {
        {  0,  1,  2 },
        { 10, 11, 12 },
        { 20, 21, 22 }};
    int *pa[3];
    for (i = 0; i < 3; i++)
        pa[i] = a[i];
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++)
            printf("%5d", pa[i][j]);
        printf("\n");
    }
    printf("%p\n", pa);
    printf("%p\n", pa[0]);
    printf("%p\n", pa[1]);
}
```

IMPLEMENTACIJA MATRICE PREKO NIZA

Nama je potrebno da imamo veću fleksibilnost, tj da se dimenzije matrice mogu uneti kao parametri našeg programa. Zbog toga je neophodno koristiti dinamičku alokaciju memorije.

```
#include <stdlib.h>
#include <stdio.h>
#define a(i, j) a[(i)*n + (j)]
int main()
{
    int m, n; /* Dimenzije matrice */
    int *a;   /* Matrica */
    int i, j;
    int s = 0; /* Suma elemenata matrice */
    /* Unos i alokacija */
    scanf("%d", &m); scanf("%d", &n);
    a = malloc(m * n * sizeof(int));
    if (a == NULL) {
        printf("Greska prilikom alokacije memorije!\n");
        return 1;
    }
```

```
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++) {
            printf("Unesi a(%d,%d) : ", i, j);
            scanf("%d", &a(i, j));
        }
    /* Racunamo sumu elemenata matrice */
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
            s += a(i, j);
    /* Ispis unete matrice */
    for (i = 0; i < m; i++) {
        for (j = 0; j < n; j++)
            printf("%d ", a(i, j));
        printf("\n");
    }
    printf("Suma elemenata matrice je %d\n", s);
    /* Oslobadjamo memoriju */
    free(a);
}
```