Vectori alocaţi dinamic

https://ocw.cs.pub.ro/courses/programare/laboratoare/lab09

Funcţii de alocare şi eliberare a memoriei

Funcțiile standard de alocare și de eliberare a memoriei sunt declarate în fişierul antet `stdlib.h`.

- `void *malloc(size_t size);`
- `void *calloc(size_t nmemb, size_t size);`
- `void *realloc(void *ptr, size_t size);`
- `void free(void *ptr);`

Exemplu

```
char *str = malloc(30);            // Aloca memorie pentru 30 de caractere
int *a = malloc(n * sizeof(int)); // Aloca memorie pt. n numere intregi
```

Alocarea de memorie pentru un vector şi iniţializarea zonei alocate cu zerouri se poate face cu funcţia `calloc`.

Exemplu

```
int *a = calloc(n, sizeof(int)); // Aloca memorie pentru n numere intregi
și inițializează zona cu zero
```

Codul de mai sus este perfect echivalent (dar mai rapid) cu următoarea secvenţă de instrucţiuni:

```
int i;
int *a = malloc(n * sizeof(int));
for (i = 0; i < n; i++) {
    a[i] = 0;
}
```

În timp ce funcţia `malloc()` ia un singur parametru (o dimensiune în octeţi), funcţia `calloc()` primeşte două argumente, o lungime de vector şi o dimensiune a fiecărui element. Astfel, această funcţie este specializată pentru memorie organizată ca un vector, în timp ce `malloc()` nu ţine cont de structura memoriei.

Exemplu

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

int main(void)
{
  int n,i;
  int *a = NULL;                    // Adresa vector alocat dinamic

  printf("n = ");
  scanf("%d", &n);              // Dimensiune vector

  a = calloc(n, sizeof(int)); // Alternativ: a = malloc(n * sizeof(int));
if (a == NULL) {
```

```
            printf("Nu s-a alocat memorie.\n");
            exit(1);
            }
  printf("Componente vector: \n");

  for (i = 0; i < n; i++) {
    scanf("%d", &a[i]);        // Sau scanf (“%d”, a+i);
  }
  for (i = 0; i < n; i++) {     // Afisare vector
    printf("%d ",a[i]);
  }

  free(a);                     // Nu uitam sa eliberam memoria

  return 0;
}
```

Matrice alocată dinamic (cu dimensiuni necunoscute la execuţie)

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

int main(void)
{
  int n, i, j;
  int **mat; // Adresa matrice

  // Citire dimensiuni matrice
  printf("n = ");
  scanf("%d", &n);

  // Alocare memorie ptr matrice
  mat = malloc(n * sizeof(int *));

  for (i = 0; i < n; i++) {
    mat[i] = calloc(n, sizeof(int));
  }

  // Completare matrice
  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
      mat[i][j] = n * i + j + 1; //*(*(mat+i)+j) = n * i + j + 1;
    }
  }

  // Afisare matrice
  for (i = 0; i < n; i++) {
    for (j = 0;j < n; j++) {
      printf("%6d", mat[i][j]); //printf("%6d", *(*(mat+i)+j));
    }
     printf("\n");
  }
   return 0;}
```