



Ministerul Educației și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Informatică și Ingineria a sistemelor

Disciplina: Programarea Calculatoarelor

RAPORT

Lucru Individual

A efectuat :

Veaceslav Mihalachi

#Probleme pentru tablouri unidimensionale:

Input

```
1 #include <stdio.h>
2
3 int main() {
4     int t[3][3];
5     int n = 3, m = 3;
6     int count[3] = {0};
7
8     printf("introduceti numere pentru matricea 3x3:\n\n");
9     for(int i = 0; i < n; i++) {
10         for(int j = 0; j < m; j++) {
11             scanf("%d", &t[i][j]);}
12
13         for(int i = 0; i < n; i++) {
14             for(int j = 0; j < m; j++) {
15                 if(t[i][j] > 0) {
16                     count[i]++;
17                 }
18
19                 for(int i = 0; i < n-1; i++) {
20                     for(int j = i+1; j < n; j++) {
21                         if(count[i] > count[j]) {
22                             int temp_c = count[i];
23                             count[i] = count[j];
24                             count[j] = temp_c;
25
26                             for(int k = 0; k < m; k++) {
27                                 int temp = t[i][k];
28                                 t[i][k] = t[j][k];
29                                 t[j][k] = temp;}}}
30
31             printf("tabloul ordonat:\n\n");
32             for(int i = 0; i < n; i++) {
33                 for(int j = 0; j < m; j++) {
34                     printf("%d ", t[i][j]);}
35                     printf("\n");}
36     return 0;
37
38 }
```

Output

```
introduceti numere pentru matricea 3x3:
{
-1 2 5
9 5 5
-4 -7 3
tabloul ordonat:

-4 -7 3
-1 2 5
9 5 5
```

Input

```
main.c
1 #include <stdio.h>
2
3 void citesteTablou(int arr[], int n, int i) {
4     if(i < n) {
5         printf("Element %d: ", i+1);
6         scanf("%d", &arr[i]);
7         citesteTablou(arr, n, i+1);
8     }
9 }
10
11 void afiseazaTablou(int arr[], int n, int i) {
12     if(i < n) {
13         printf("%d ", arr[i]);
14         afiseazaTablou(arr, n, i+1);
15     }
16 }
17
18 int main() {
19     int n;
20     printf("Numar elemente: ");
21     scanf("%d", &n);
22
23     int arr[n];
24     citesteTablou(arr, n, 0);
25
26     printf("Tabloul: ");
27     afiseazaTablou(arr, n, 0);
28
29     return 0;
30 }
```

Output

```
Numar elemente: 3
Element 1: 2
Element 2: 4
Element 3: 1
Tabloul: 2 4 1
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9     printf("Introduceti %d elemente:\n", n);
10    for(int i = 0; i < n; i++) {
11        scanf("%d", &arr[i]);
12    }
13
14    printf("Elemente negative: ");
15    for(int i = 0; i < n; i++) {
16        if(arr[i] < 0) {
17            printf("%d ", arr[i]);
18        }
19    }
20
21    return 0;
22 }
```

Output

```
Numar elemente: 3
Introduceti 3 elemente:
-4
2
1
Elemente negative: -4
```

Input

```
main.c
1 #include <stdio.h>
2
3 int sumaRecursiv(int arr[], int n) {
4     if(n <= 0) return 0;
5     return arr[n-1] + sumaRecursiv(arr, n-1);
6 }
7
8 int main() {
9     int n;
10    printf("Numar elemente: ");
11    scanf("%d", &n);
12
13    int arr[n];
14    printf("Introduceti %d elemente:\n", n);
15    for(int i = 0; i < n; i++) {
16        scanf("%d", &arr[i]);
17    }
18
19    printf("Suma: %d\n", sumaRecursiv(arr, n));
20
21    return 0;
22 }
```

Output

```
Numar elemente: 4
Introduceti 4 elemente:
13
-2
3
7
Suma: 21
```

Input

```
main.c
1 #include <stdio.h>
2 #include <limits.h>
3
4 int maximRecursiv(int arr[], int n) {
5     if(n == 1) return arr[0];
6
7     int max_rest = maximRecursiv(arr, n-1);
8     return (arr[n-1] > max_rest) ? arr[n-1] : max_rest;
9 }
10
11 int minimRecursiv(int arr[], int n) {
12     if(n == 1) return arr[0];
13
14     int min_rest = minimRecursiv(arr, n-1);
15     return (arr[n-1] < min_rest) ? arr[n-1] : min_rest;
16 }
17
18 int main() {
19     int n;
20     printf("Numar elemente: ");
21     scanf("%d", &n);
22
23     int arr[n];
24     printf("Introduceti %d elemente:\n", n);
25     for(int i = 0; i < n; i++) {
26         scanf("%d", &arr[i]);
27     }
28
29     printf("Maxim: %d\n", maximRecursiv(arr, n));
30     printf("Minim: %d\n", minimRecursiv(arr, n));
31
32     return 0;
33 }
```

Output

```
Numar elemente: 3
Introduceti 3 elemente:
2
5
3
Maxim: 5
Minim: 2
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9     printf("Introduceti %d elemente:\n", n);
10    for(int i = 0; i < n; i++) {
11        scanf("%d", &arr[i]);
12    }
13
14    int pare = 0, impare = 0;
15    for(int i = 0; i < n; i++) {
16        if(arr[i] % 2 == 0) pare++;
17        else impare++;
18    }
19
20    printf("Pare: %d\nImpare: %d\n", pare, impare);
21
22    return 0;
23 }
```

Output

```
Numar elemente: 3
Introduceti 3 elemente:
3
7
10
Pare: 1
Impare: 2
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9     printf("Introduceti %d elemente:\n", n);
10    for(int i = 0; i < n; i++) {
11        scanf("%d", &arr[i]);
12    }
13
14    int negative = 0;
15    for(int i = 0; i < n; i++) {
16        if(arr[i] < 0) negative++;
17    }
18
19    printf("Elemente negative: %d\n", negative);
20
21    return 0;
22 }
```

Output

```
Introduceti 5 elemente:
-4
-2
1
5
-8
Elemente negative: 3
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int sursa[n];
9     int destinatie[n];
10
11    printf("Introduceti %d elemente:\n", n);
12    for(int i = 0; i < n; i++) {
13        scanf("%d", &sursa[i]);
14    }
15
16    for(int i = 0; i < n; i++) {
17        destinatie[i] = sursa[i];
18    }
19
20    printf("Tabloul copiat: ");
21    for(int i = 0; i < n; i++) {
22        printf("%d ", destinatie[i]);
23    }
24
25    return 0;
26 }
```

Output

```
Numar elemente: 3
Introduceti 3 elemente:
4
-1
87
Tabloul copiat: 4 -1 87
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente initiale: ");
6     scanf("%d", &n);
7
8     int arr[n+1];
9
10    printf("Introduceti %d elemente:\n", n);
11   for(int i = 0; i < n; i++) {
12       scanf("%d", &arr[i]);
13   }
14
15    int valoare, pozitie;
16    printf("Valoare de inserat: ");
17    scanf("%d", &valoare);
18    printf("Pozitie (0-%d): ", n);
19    scanf("%d", &pozitie);
20   for(int i = n; i > pozitie; i--) {
21       arr[i] = arr[i-1];
22   }
23   arr[pozitie] = valoare;
24
25   printf("Tabloul dupa inserare: ");
26   for(int i = 0; i <= n; i++) {
27       printf("%d ", arr[i]);
28   }
29
30   return 0;
31 }
```

Output

```
Numar elemente initiale: 4
Introduceti 4 elemente:
20
10
4
3
Valoare de inserat: 2
Pozitie (0-4): 1
Tabloul dupa inserare: 20 2 10 4 3
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    int pozitie;
16    printf("Pozitie de sters (0-%d): ", n-1);
17    scanf("%d", &pozitie);
18    for(int i = pozitie; i < n-1; i++) {
19        arr[i] = arr[i+1];
20    }
21
22    printf("Tabloul dupa stergere: ");
23    for(int i = 0; i < n-1; i++) {
24        printf("%d ", arr[i]);
25    }
26
27    return 0;
28 }
```

Output

```
Numar elemente: 3
Introduceti 3 elemente:
13
-25
90
Pozitie de sters (0-2): 2
Tabloul dupa stergere: 13 -25
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    printf("Frecventa elementelor:\n");
16
17    for(int i = 0; i < n; i++) {
18        if(arr[i] == -1) continue;
19
20        int count = 1;
21        for(int j = i+1; j < n; j++) {
22            if(arr[i] == arr[j]) {
23                count++;
24                arr[j] = -1;
25            }
26        }
27        printf("%d apare de %d ori\n", arr[i], count);
28    }
29    return 0;
30 }
```

Output

```
11
3
11
24
Frecventa elementelor:
11 apare de 2 ori
3 apare de 1 ori
24 apare de 1 ori
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    printf("Elemente unice: ");
16
17    for(int i = 0; i < n; i++) {
18        int esteUnic = 1;
19        for(int j = 0; j < n; j++) {
20            if(i != j && arr[i] == arr[j]) {
21                esteUnic = 0;
22                break;
23            }
24        }
25        if(esteUnic) {
26            printf("%d ", arr[i]);
27        }
28    }
29
30    return 0;
31 }
```

Output

```
Numar elemente: 4
Introduceti 4 elemente:
13
13
2
-4
Elemente unice: 2 -4
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    int duplicate = 0;
16
17    for(int i = 0; i < n; i++) {
18        if(arr[i] == -1) continue;
19
20        int count = 1;
21        for(int j = i+1; j < n; j++) {
22            if(arr[i] == arr[j]) {
23                count++;
24                arr[j] = -1;
25            }
26        }
27        if(count > 1) {
28            duplicate += (count - 1);
29        }
30    }
31
32    printf("Numar total de duplicate: %d\n", duplicate);
33
34    return 0;
35 }
```

Output

```
Introduceti 3 elemente:
25
25
1
Numar total de duplicate: 1

...Program finished with exit code 0
Press ENTER to exit console.
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    for(int i = 0; i < n; i++) {
16        for(int j = i+1; j < n; j++) {
17            if(arr[i] == arr[j]) {
18                for(int k = j; k < n-1; k++) {
19                    arr[k] = arr[k+1];
20                }
21                n--;
22                j--;
23            }
24        }
25    }
26
27    printf("Tablou fara duplicate: ");
28    for(int i = 0; i < n; i++) {
29        printf("%d ", arr[i]);
30    }
31
32    return 0;
33 }
```

Output

```
Numar elemente: 4
Introduceti 4 elemente:
13
-2
24
13
Tablou fara duplicate: 13 -2 24
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n1, n2;
5
6     printf("Dimensiune primul tablou: ");
7     scanf("%d", &n1);
8     int arr1[n1];
9     printf("Introduceti %d elemente:\n", n1);
10    for(int i = 0; i < n1; i++) {
11        scanf("%d", &arr1[i]);
12    }
13
14    printf("Dimensiune al doilea tablou: ");
15    scanf("%d", &n2);
16    int arr2[n2];
17    printf("Introduceti %d elemente:\n", n2);
18    for(int i = 0; i < n2; i++) {
19        scanf("%d", &arr2[i]);
20    }
21
22    int arr3[n1 + n2];
23    for(int i = 0; i < n1; i++) {
24        arr3[i] = arr1[i];
25    }
26    for(int i = 0; i < n2; i++) {
27        arr3[n1 + i] = arr2[i];
28    }
29
30    printf("Tabloul imbunatit: ");
31    for(int i = 0; i < n1 + n2; i++) {
32        printf("%d ", arr3[i]);
33    }
34
35    return 0;
36 }
```

Output

```
13
-2
24
13
Tablou fara duplicate: 13 -2 24
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14    for(int i = 0; i < n/2; i++) {
15        int temp = arr[i];
16        arr[i] = arr[n-i-1];
17        arr[n-i-1] = temp;
18    }
19
20    printf("Tablou inversat: ");
21    for(int i = 0; i < n; i++) {
22        printf("%d ", arr[i]);
23    }
24
25    return 0;
26 }
```

Output

```
Introduceti 3 elemente:
13
19
2
Tablou inversat: 2 19 13
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14    int pare[n], impare[n];
15    int count_pare = 0, count_impare = 0;
16
17    for(int i = 0; i < n; i++) {
18        if(arr[i] % 2 == 0) {
19            pare[count_pare++] = arr[i];
20        } else {
21            impare[count_impare++] = arr[i];
22        }
23    }
24    printf("Elemente pare: ");
25    for(int i = 0; i < count_pare; i++) {
26        printf("%d ", pare[i]);
27    }
28    printf("\nElemente impare: ");
29    for(int i = 0; i < count_impare; i++) {
30        printf("%d ", impare[i]);
31    }
32
33    return 0;
34 }
```

Output

```
Introduceti 3 elemente:
3
15
10
Elemente pare: 10
Elemente impare: 3 15
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7
8     int arr[n];
9
10    printf("Introduceti %d elemente:\n", n);
11    for(int i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    int element;
16    printf("Element de cautat: ");
17    scanf("%d", &element);
18
19    printf("Pozitii gasite: ");
20    int gasit = 0;
21
22    for(int i = 0; i < n; i++) {
23        if(arr[i] == element) {
24            printf("%d ", i);
25            gasit = 1;
26        }
27    }
28
29    if(!gasit) {
30        printf("Elementul nu a fost gasit!");
31    }
32
33    return 0;
34 }
```

Output

```
Introduceti 4 elemente:
14
18
21
28
Element de cautat: 21
Pozitii gasite: 2
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Numar elemente: ");
6     scanf("%d", &n);
7     int arr[n];
8     printf("Introduceti %d elemente:\n", n);
9     for(int i = 0; i < n; i++) {
10         scanf("%d", &arr[i]);
11     }
12     int alegere;
13     printf("1 - Crescator\n2 - Descrescator\nAlegere: ");
14     scanf("%d", &alegere);
15     for(int i = 0; i < n-1; i++) {
16         for(int j = 0; j < n-i-1; j++) {
17             if((alegere == 1 && arr[j] > arr[j+1]) ||
18                 (alegere == 2 && arr[j] < arr[j+1])) {
19                 int temp = arr[j];
20                 arr[j] = arr[j+1];
21                 arr[j+1] = temp;
22             }
23         }
24     }
25     printf("Tablou sortat: ");
26     for(int i = 0; i < n; i++) {
27         printf("%d ", arr[i]);
28     }
29
30     return 0;
31 }
```

Output

```
-2
24
37
11
9
1 - Crescator
2 - Descrescator
Alegere: 1
Tablou sortat: -2 9 11 24 37
```

#Probleme pentru tablouri bidimensionale:

```

main.c

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 int** alocaMatrice(int n, int m) {
5     int **matrix = (int **)malloc(n * sizeof(int *));
6     if (matrix == NULL) {
7         printf("Eroare la alocarea memoriei pentru linii.\n");
8         return NULL;
9     }
10    for (int i = 0; i < n; i++) {
11        matrix[i] = (int *)malloc(m * sizeof(int));
12        if (matrix[i] == NULL) {
13            printf("Eroare la alocarea memoriei pentru coloane.\n");
14            for (int j = 0; j < i; j++) {
15                free(matrix[j]);
16            }
17            free(matrix);
18            return NULL;
19    }
20    return matrix;
21}
22 void elibereazaMatrice(int **matrix, int n) {
23    if (matrix != NULL) {
24        for (int i = 0; i < n; i++) {
25            free(matrix[i]);
26        }
27        free(matrix);
28    }
29}
30 void citesteMatrice(int **matrix, int n, int m) {
31    printf("Introduceti elementele matricei (%d linii x %d coloane):\n", n, m);
32    for (int i = 0; i < n; i++) {
33        for (int j = 0; j < m; j++) { printf("", i, j);
34            scanf("%d", &matrix[i][j]);}
35    }
36}
37 void completeazaAleator(int **matrix, int n, int m) {
38    srand((time(NULL)));
39    for (int i = 0; i < n; i++) {
40        for (int j = 0; j < m; j++) {
41            matrix[i][j] = rand() % 100;}
42    }
43    printf("Matricea a fost completata cu valori aleatorii!\n");
44}
45 void insertionSort(int v[], int size) {
46    for (int i = 1; i < size; i++) {
47        int key = v[i];
48        int j = i - 1;
49        while (j >= 0 && v[j] > key) {
50            v[j + 1] = v[j];
51            j--;
52        }
53        v[j + 1] = key;
54    }
55}
56 void sorteazaPerimetru(int **matrix, int n, int m) {
57    int size = 2*(n + m) - 4;
58    int *v = (int *)malloc(size * sizeof(int));
59    int k = 0;
60    for (int j = 0; j < m; j++) v[k++] = matrix[0][j];
61    for (int i = 1; i < n-1; i++) v[k++] = matrix[i][m-1];
62    for (int j = m-1; j >= 0; j--) v[k++] = matrix[n-1][j];
63    for (int i = n-2; i > 0; i--) v[k++] = matrix[i][0];
64    insertionSort(v, size);
65}

```

```

main.c
45     v[j+ 1] = key;}}
46 void sorteazaPerimetru(int **matrix, int n, int m) {
47     int size = 2*(n + m) - 4;
48     int *v = (int*)malloc(size * sizeof(int));
49     int k = 0;
50     for (int j = 0; j < m; j++) v[k++] = matrix[0][j];
51     for (int i = 1; i < n-1; i++) v[k++] = matrix[i][m-1];
52     for (int j = m-1; j >= 0; j--) v[k++] = matrix[n-1][j];
53     for (int i = n-2; i > 0; i--) v[k++] = matrix[i][0];
54     insertionSort(v, size);
55     k = 0;
56     for (int j = 0; j < m; j++) matrix[0][j] = v[k++];
57     for (int i = 1; i < n-1; i++) matrix[i][m-1] = v[k++];
58     for (int j = m-1; j >= 0; j--) matrix[n-1][j] = v[k++];
59     for (int i = n-2; i > 0; i--) matrix[i][0] = v[k++];
60     free(v);
61     printf("Perimetrul matricei a fost sortat ascendent.\n");
62
63 void afiseazaMatrice(int **matrix, int n, int m) {
64     if (matrix == NULL) {
65         printf("Matricea nu este alocată.\n");
66         return;
67     }
68     printf("Elementele matricei:\n");
69     for (int i = 0; i < n; i++) {
70         for (int j = 0; j < m; j++) {
71             printf("%d ", matrix[i][j]);
72         }
73         printf("\n");
74     }
75     int n = 0, m = 0;
76     int **matrix = NULL;
77     int optiune;
78
79     do {
80         printf("\n                  [Meniu]\n");
81         printf("1. Alocarea dinamica a memoriei pentru tabloul bidimensional\n");
82         printf("2. Introducerea elementelor tabloului de la tastatura\n");
83         printf("3. Completarea tabloului cu valori aleatorii\n");
84         printf("4. Sortarea elementelor de pe perimetru matricei\n");
85         printf("5. Afisarea elementelor tabloului la ecran\n");
86         printf("6. Eliberarea memoriei alocate pentru tablou\n");
87         printf("0. Iesire din program\n");printf("Alege o optiune: ");
88         scanf("%d", &optiune);
89     }

```

```

main.c
89     switch (optiune){
90         case 1:
91             if (matrix != NULL) {
92                 printf("Matricea este deja alocată. Eliberez-o mai întai.\n");}else {
93                 printf("Introduceți numărul de linii : ");
94                 scanf("%d", &n);
95                 printf("Introduceți numărul de coloane : ");
96                 scanf("%d", &m);
97                 matrix = alocaMatrice(n, m);
98                 if (matrix!= NULL) {
99                     printf("Memoria a fost alocată cu succes.\n");} break;
100            case 2:
101                if (matrix == NULL) {
102                    printf("Matricea nu e alocată. Alege optiunea 1 mai întai.\n");} else {
103                        citesteMatrice(matrix, n, m);}
104                        break;
105            case 3:
106                if (matrix ==NULL) {
107                    printf("Matricea nu este alocată. Alege optiunea 1 mai întai.\n");} else {
108                        completeazaAleator(matrix, n, m);}
109                        break;
110            case 4:
111                if (matrix == NULL) {
112                    printf("Matricea nu este alocată. Alege optiunea 1 mai întai.\n");} else {
113                        sorteazaPerimetru(matrix, n, m);}
114                        break;
115            case 5:
116                afiseazaMatrice(matrix,n,m);
117                break;
118            case 6:
119                elibereazaMatrice(matrix,n);
120                matrix = NULL;n = 0;m =0;
121                printf("Memoria a fost eliberata\n");
122                break;
123            case 0:
124                if (matrix != NULL) {
125                    elibereazaMatrice(matrix, n);
126                    }
127                    printf("Ieșire din program\n");
128                    break;
129                    default:
130                        printf("Optiune invalidă. Încearcă din nou!!\n");
131                        break;
132        } while (optiune != 0);
133        return 0;

```

Output

```
|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 1
Introduceți numărul de linii : 3
Introduceți numărul de coloane : 3
Memoria a fost alocată cu succes.

|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 2
Introduceți elementele matricei (3 linii x 3 coloane):
3 65 39
4 1 -3
53 9 2

|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 4
Perimetrul matricei a fost sortat ascendent.

|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 5
Elementele matricei:
-3 2 3
65 1 4
53 39 9
```

```
|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 5
Elementele matricei:
-3 2 3
65 1 4
53 39 9

|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 6
Memoria a fost eliberata

|Meniu|
1. Alocarea dinamica a memoriei pentru tabloul bidimensional
2. Introducerea elementelor tabloului de la tastatura
3. Completarea tabloului cu valori aleatorii
4. Sortarea elementelor de pe perimetru matricei
5. Afisarea elementelor tabloului la ecran
6. Eliberarea memoriei alocate pentru tablou
0. Iesire din program
Alege o optiune: 0
Iesire din program

...Program finished with exit code 0
Press ENTER to exit console.
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int A[r][c], B[r][c], suma[r][c];
9     printf("Introduceti matricea A:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &A[i][j]);
13        }
14    }
15    printf("Introduceti matricea B:\n");
16    for(int i = 0; i < r; i++) {
17        for(int j = 0; j < c; j++) {
18            scanf("%d", &B[i][j]);
19        }
20    }
21    for(int i = 0; i < r; i++) {
22        for(int j = 0; j < c; j++) {
23            suma[i][j] = A[i][j] + B[i][j];
24        }
25    }
26    printf("Matricea suma:\n");
27    for(int i = 0; i < r; i++) {
28        for(int j = 0; j < c; j++) {
29            printf("%d ", suma[i][j]);
30        }
31        printf("\n");
32    }
33    return 0;
34 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea A:
3 12
34 3
Introduceti matricea B:
12 2
34 1
Matricea suma:
15 14
68 4
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int A[r][c], B[r][c], diferenta[r][c];
9     printf("Introduceti matricea A:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &A[i][j]);
13        }
14    }
15    printf("Introduceti matricea B:\n");
16    for(int i = 0; i < r; i++) {
17        for(int j = 0; j < c; j++) {
18            scanf("%d", &B[i][j]);
19        }
20    }
21    for(int i = 0; i < r; i++) {
22        for(int j = 0; j < c; j++) {
23            diferenta[i][j] = A[i][j] - B[i][j];
24        }
25    }
26    printf("Matricea diferenta (A - B):\n");
27    for(int i = 0; i < r; i++) {
28        for(int j = 0; j < c; j++) {
29            printf(" %d ", diferenta[i][j]);
30        }
31        printf("\n");
32    }
33    return 0;
34 }
```

Output

```
-
Numar randuri: 2
Numar coloane: 2
Introduceti matricea A:
13 29
-43 3
Introduceti matricea B:
-11 -24
16 19
Matricea diferenta (A - B):
24 53
-59 -16
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int r, c, scalar;
5     printf("Numar randuri: ");
6     scanf("%d", &r);
7     printf("Numar coloane: ");
8     scanf("%d", &c);
9     int A[r][c], rezultat[r][c];
10    printf("Introduceti matricea:\n");
11    for(int i = 0; i < r; i++) {
12        for(int j = 0; j < c; j++) {
13            scanf("%d", &A[i][j]);
14        }
15    }
16    printf("Introduceti scalarul: ");
17    scanf("%d", &scalar);
18    for(int i = 0; i < r; i++) {
19        for(int j = 0; j < c; j++) {
20            rezultat[i][j] = A[i][j] * scalar;
21        }
22    }
23    printf("Matricea dupa multiplicare cu %d:\n", scalar);
24    for(int i = 0; i < r; i++) {
25        for(int j = 0; j < c; j++) {
26            printf("%d ", rezultat[i][j]);
27        }
28        printf("\n");
29    }
30
31    return 0;
32 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
34 1
24 5
Introduceti scalarul: 5
Matricea dupa multiplicare cu 5:
170 5
120 25
```

Input

```

main.c
1 #include <stdio.h>
2 int main() {
3     int r1, c1, r2, c2;
4     printf("Matricea A:\n");
5     printf("Randuri: ");
6     scanf("%d", &r1);
7     printf("Coloane: ");
8     scanf("%d", &c1);
9     printf("Matricea B:\n");
10    printf("Randuri: ");
11    scanf("%d", &r2);
12    printf("Coloane: ");
13    scanf("%d", &c2);
14    if(c1 != r2) {
15        printf("Nu se pot multiplica! (c1 != r2)\n");
16        return 0;
17    }
18    int A[r1][c1], B[r2][c2], C[r1][c2];
19    printf("Introduceti matricea A:\n");
20    for(int i = 0; i < r1; i++) {
21        for(int j = 0; j < c1; j++) {
22            scanf("%d", &A[i][j]);
23        }
24    }
25    printf("Introduceti matricea B:\n");
26    for(int i = 0; i < r2; i++) {
27        for(int j = 0; j < c2; j++) {
28            scanf("%d", &B[i][j]);
29        }
30        for(int i = 0; i < r1; i++) {
31            for(int j = 0; j < c2; j++) {
32                for(int k = 0; k < c1; k++) {
33                    C[i][j] += A[i][k] * B[k][j];
34                }
35            }
36        }
37        printf("\n");
38    }
39    return 0;
40 }

```

Output

```

Matricea A:
Randuri: 2
Coloane: 2
Matricea B:
Randuri: 2
Coloane: 2
Introduceti matricea A:
-13 -5
2 31
Introduceti matricea B:
2 5
7 3
Matricea rezultat:
-61 -80
221 103

```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int A[r][c], B[r][c];
9     printf("Introduceti matricea A:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &A[i][j]);
13        }
14    }
15    printf("Introduceti matricea B:\n");
16    for(int i = 0; i < r; i++) {
17        for(int j = 0; j < c; j++) {
18            scanf("%d", &B[i][j]);
19        }
20    }
21    int egale = 1;
22    for(int i = 0; i < r; i++) {
23        for(int j = 0; j < c; j++) {
24            if(A[i][j] != B[i][j]) {
25                egale = 0;
26                break;
27            }
28        }
29        if(!egale) break;
30    }
31    if(egale) {
32        printf("Matricele sunt egale.\n");
33    } else {
34        printf("Matricele NU sunt egale.\n");
35    }
36    return 0;
37 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea A:
2 3
3 2
Introduceti matricea B:
2 3
3 2
Matricele sunt egale.
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica (n x n): ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int suma = 0;
18    for(int i = 0; i < n; i++) {
19        suma += mat[i][i];
20    }
21
22    printf("Suma diagonalei principale: %d\n", suma);
23
24    return 0;
25 }
```

Output

```
Dimensiune matrice patratica: 2
Introduceti matricea 2x2:
2 4
3 1
Suma diagonalei principale: 3
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica: ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int suma = 0;
18    for(int i = 0; i < n; i++) {
19        suma += mat[i][n-1-i];
20    }
21
22    printf("Suma diagonalei secundare: %d\n", suma);
23
24    return 0;
25 }
```

Output

```
Dimensiune matrice patratica: 2
Introduceti matricea 2x2:
3 5
1 7
Suma diagonalei secundare: 6
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int mat[r][c];
9     printf("Introduceti matricea:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &mat[i][j]);
13        }
14    }
15    printf("Suma randuri:\n");
16    for(int i = 0; i < r; i++) {
17        int suma_rand = 0;
18        for(int j = 0; j < c; j++) {
19            suma_rand += mat[i][j];
20        }
21        printf("Rand %d: %d\n", i+1, suma_rand);
22    }
23    printf("\nSuma coloane:\n");
24    for(int j = 0; j < c; j++) {
25        int suma_coloana = 0;
26        for(int i = 0; i < r; i++) {
27            suma_coloana += mat[i][j];
28        }
29        printf("Coloana %d: %d\n", j+1, suma_coloana);
30    }
31    return 0;
32 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
23 25
51 2
Suma randuri:
Rand 1: 48
Rand 2: 53

Suma coloane:
Coloana 1: 74
Coloana 2: 27
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int n;
4     printf("Dimensiune matrice patratica (n x n): ");
5     scanf("%d", &n);
6     int mat[n][n];
7     printf("Introduceti matricea %dx%d:\n", n, n);
8     for(int i = 0; i < n; i++) {
9         for(int j = 0; j < n; j++) {
10            scanf("%d", &mat[i][j]);
11        }
12    }
13    for(int i = 0; i < n; i++) {
14        int temp = mat[i][i];
15        mat[i][i] = mat[i][n-1-i];
16        mat[i][n-1-i] = temp;
17    }
18    printf("Matricea dupa schimbarea diagonalelor:\n");
19    for(int i = 0; i < n; i++) {
20        for(int j = 0; j < n; j++) {
21            printf("%d ", mat[i][j]);
22        }
23        printf("\n");
24    }
25
26    return 0;
27 }
```

Output

```
Dimensiune matrice patratica: 2
Introduceti matricea 2x2:
2 5
6 1
Matricea dupa schimbarea diagonalelor:
5 2
1 6
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int n;
4     printf("Dimensiune matrice patratica: ");
5     scanf("%d", &n);
6     int mat[n][n];
7     printf("Introduceti matricea %dx%d:\n", n, n);
8     for(int i = 0; i < n; i++) {
9         for(int j = 0; j < n; j++) {
10            scanf("%d", &mat[i][j]);
11        }
12    }
13    printf("Matricea triunghiulara superioara:\n");
14    for(int i = 0; i < n; i++) {
15        for(int j = 0; j < n; j++) {
16            if(i <= j) {
17                printf("%d ", mat[i][j]);
18            } else {
19                printf("0 ");
20            }
21        }
22        printf("\n");
23    }
24
25    return 0;
26 }
```

Output

```
Dimensiune matrice patratica): 2
Introduceti matricea 2x2:
42 1
54 6
Matricea triunghiulara superioara:
42 1
0 6
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica: ");
6     scanf("%d", &n);
7     int mat[n][n];
8     printf("Introduceti matricea %dx%d:\n", n, n);
9     for(int i = 0; i < n; i++) {
10        for(int j = 0; j < n; j++) {
11            scanf("%d", &mat[i][j]);
12        }
13    }
14    printf("Matricea triunghiulara inferioara:\n");
15    for(int i = 0; i < n; i++) {
16        for(int j = 0; j < n; j++) {
17            if(i >= j) {
18                printf("%d ", mat[i][j]);
19            } else {
20                printf("0 ");
21            }
22        }
23        printf("\n");
24    }
25
26    return 0;
27 }
```

Output

```
Dimensiune matrice patratica: 2
Introduceti matricea 2x2:
23 7
1 23
Matricea triunghiulara inferioara:
23 0
1 23
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica: ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int suma = 0;
18    for(int i = 0; i < n; i++) {
19        for(int j = i; j < n; j++) {
20            suma += mat[i][j];
21        }
22    }
23
24    printf("Suma triunghiulara superioara: %d\n", suma);
25
26    return 0;
27 }
```

Output

```
Dimensiune matrice patratica: 2
Introduceti matricea 2x2:
1 2
3 4
Suma triunghiulara superioara: 7
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica |: ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int suma = 0;
18    for(int i = 0; i < n; i++) {
19        for(int j = 0; j <= i; j++) {
20            suma += mat[i][j];
21        }
22    }
23
24    printf("Suma triunghiulara inferioara: %d\n", suma);
25
26    return 0;
27 }
```

Output

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica |: ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int suma = 0;
18    for(int i = 0; i < n; i++) {
19        for(int j = 0; j <= i; j++) {
20            suma += mat[i][j];
21        }
22    }
23
24    printf("Suma triunghiulara inferioara: %d\n", suma);
25
26    return 0;
27 }
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int mat[r][c], transpusa[c][r];
9     printf("Introduceti matricea:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &mat[i][j]);
13        }
14    }
15
16    for(int i = 0; i < r; i++) {
17        for(int j = 0; j < c; j++) {
18            transpusa[j][i] = mat[i][j];
19        }
20    }
21    printf("Matricea transpusa:\n");
22    for(int i = 0; i < c; i++) {
23        for(int j = 0; j < r; j++) {
24            printf("%d ", transpusa[i][j]);
25        }
26        printf("\n");
27    }
28
29    return 0;
30 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
2 4
1 55
Matricea transpusa:
2 1
4 55
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice: ");
6     scanf("%d", &n);
7
8     if(n != 2 && n != 3) {
9         printf("\n");
10        return 0;
11    }
12
13    int mat[n][n];
14
15    printf("Introduceti matricea %dx%d:\n", n, n);
16    for(int i = 0; i < n; i++) {
17        for(int j = 0; j < n; j++) {
18            scanf("%d", &mat[i][j]);
19        }
20    }
21
22    int det;
23    if(n == 2) {
24        det = mat[0][0] * mat[1][1] - mat[0][1] * mat[1][0];
25    } else {
26        det = mat[0][0]*mat[1][1]*mat[2][2] +
27              mat[0][1]*mat[1][2]*mat[2][0] +
28              mat[0][2]*mat[1][0]*mat[2][1] -
29              mat[0][2]*mat[1][1]*mat[2][0] -
30              mat[0][0]*mat[1][2]*mat[2][1] -
31              mat[0][1]*mat[1][0]*mat[2][2];
32    }
33
34    printf("Determinantul: %d\n", det);
35
36    return 0;
37 }
```

Output

```
1 7
Determinantul: 16
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica : ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int esteUnitar = 1;
18
19    for(int i = 0; i < n; i++) {
20        for(int j = 0; j < n; j++) {
21            if(i == j) {
22                if(mat[i][j] != 1) esteUnitar = 0;
23            } else {
24                if(mat[i][j] != 0) esteUnitar = 0;
25            }
26        }
27    }
28
29    if(esteUnitar) {
30        printf("Matricea este unitara (matrice identitate).\n");
31    } else {
32        printf("Matricea NU este unitara.\n");
33    }
34
35    return 0;
36 }
```

Output

```
Dimensiune matrice patratica : 2
Introduceti matricea 2x2:
3 5
1 7
Matricea NU este unitara.
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int mat[r][c];
9     printf("Introduceti matricea:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &mat[i][j]);
13        }
14    }
15
16    int zerouri = 0;
17    int total = r * c;
18    for(int i = 0; i < r; i++) {
19        for(int j = 0; j < c; j++) {
20            if(mat[i][j] == 0) {
21                zerouri++;
22            }
23        }
24    }
25
26    float procent = (float)zerouri / total * 100;
27    printf("Zerouri: %d/%d (%.2f%%)\n", zerouri, total, procent);
28
29    if(procent >= 70) {
30        printf("Matricea este rara (sparse).\n");
31    } else {
32        printf("Matricea NU este rara.\n");
33    }
34
35    return 0;
36 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
0 65
0 43
Zerouri: 2/4 (50.00%)
Matricea NU este rara.
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     printf("Dimensiune matrice patratica: ");
6     scanf("%d", &n);
7
8     int mat[n][n];
9
10    printf("Introduceti matricea %dx%d:\n", n, n);
11    for(int i = 0; i < n; i++) {
12        for(int j = 0; j < n; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16
17    int simetrica = 1;
18    for(int i = 0; i < n; i++) {
19        for(int j = 0; j < n; j++) {
20            if(mat[i][j] != mat[j][i]) {
21                simetrica = 0;
22                break;
23            }
24        }
25        if(!simetrica) break;
26    }
27    if(simetrica) {
28        printf("Matricea este simetrica.\n");
29    } else {
30        printf("Matricea NU este simetrica.\n");
31    }
32    return 0;
33 }
```

Output

```
Dimensiune matrice patratica: 2
Introduceti matricea 2x2:
32 54
22 12
Matricea NU este simetrica.
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int r, c;
5     printf("Numar randuri: ");
6     scanf("%d", &r);
7     printf("Numar coloane: ");
8     scanf("%d", &c);
9
10    int mat[r][c];
11    printf("Introduceti matricea:\n");
12    for(int i = 0; i < r; i++) {
13        for(int j = 0; j < c; j++) {
14            scanf("%d", &mat[i][j]);
15        }
16    }
17    for(int i = 0; i < r; i++) {
18        int temp = mat[i][0];
19        for(int j = 0; j < c-1; j++) {
20            mat[i][j] = mat[i][j+1];
21        }
22        mat[i][c-1] = temp;
23    }
24    printf("Matricea dupa ROL:\n");
25    for(int i = 0; i < r; i++) {
26        for(int j = 0; j < c; j++) {
27            printf("%d ", mat[i][j]);
28        }
29        printf("\n");
30    }
31    return 0;
32 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
22 44
31 1
Matricea dupa ROL:
44 22
1 31
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int mat[r][c];
9     printf("Introduceti matricea:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &mat[i][j]);
13        }
14    }
15    for(int i = 0; i < r; i++) {
16        int temp = mat[i][c-1];
17        for(int j = c-1; j > 0; j--) {
18            mat[i][j] = mat[i][j-1];
19        }
20        mat[i][0] = temp;
21    }
22    printf("Matricea dupa ROR:\n");
23    for(int i = 0; i < r; i++) {
24        for(int j = 0; j < c; j++) {
25            printf("%d ", mat[i][j]);
26        }
27        printf("\n");
28    }
29
30    return 0;
31 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
13 15
23 22
Matricea dupa ROR:
15 13
22 23
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int mat[r][c];
9     printf("Introduceti matricea:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &mat[i][j]);
13        }
14    }
15    int temp[c];
16    for(int j = 0; j < c; j++) {
17        temp[j] = mat[0][j];
18    }
19    for(int i = 0; i < r-1; i++) {
20        for(int j = 0; j < c; j++) {
21            mat[i][j] = mat[i+1][j];
22        }
23    }
24    for(int j = 0; j < c; j++) {
25        mat[r-1][j] = temp[j];
26    }
27    printf("Matricea dupa rotire sus:\n");
28    for(int i = 0; i < r; i++) {
29        for(int j = 0; j < c; j++) {
30            printf("%d ", mat[i][j]);
31        }
32        printf("\n");
33    }
34    return 0;
35 }
```

Output

```
Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
24 26
35 12
Matricea dupa rotire sus:
35 12
24 26
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     int r, c;
4     printf("Numar randuri: ");
5     scanf("%d", &r);
6     printf("Numar coloane: ");
7     scanf("%d", &c);
8     int mat[r][c];
9     printf("Introduceti matricea:\n");
10    for(int i = 0; i < r; i++) {
11        for(int j = 0; j < c; j++) {
12            scanf("%d", &mat[i][j]);
13        }
14    }
15
16    int temp[c];
17    for(int j = 0; j < c; j++) {
18        temp[j] = mat[r-1][j];
19    }
20    for(int i = r-1; i > 0; i--) {
21        for(int j = 0; j < c; j++) {
22            mat[i][j] = mat[i-1][j];
23        }
24    }
25    for(int j = 0; j < c; j++) {
26        mat[0][j] = temp[j];
27    }
28
29    printf("Matricea dupa rotire jos:\n");
30    for(int i = 0; i < r; i++) {
31        for(int j = 0; j < c; j++) {
32            printf("%d ", mat[i][j]);
33        }
34        printf("\n");
35    }
36
37    return 0;
38 }
```

Output

```
Numar coloane: 3
Introduceti matricea:
20 39 11
23 1 9
23 54 23
Matricea dupa rotire jos:
23 54 23
20 39 11
23 1 9
```

Input

```

main.c
1 #include <stdio.h>
2
3 int main() {
4     int r, c;
5     printf("Numar randuri: ");
6     scanf("%d", &r);
7     printf("Numar coloane: ");
8     scanf("%d", &c);
9     int mat[r][c];
10    printf("Introduceti matricea:\n");
11    for(int i = 0; i < r; i++) {
12        for(int j = 0; j < c; j++) {
13            scanf("%d", &mat[i][j]);
14        }
15    }
16    int pool_size;
17    printf("Dimensiune fereastra pooling : ");
18    scanf("%d", &pool_size);
19    int rez_r = r / pool_size;
20    int rez_c = c / pool_size;
21    if(rez_r == 0 || rez_c == 0) {
22        printf("Fereastra prea mare pentru aceasta matrice!\n");
23        return 0;
24    }
25
26    int rezultat[rez_r][rez_c];
27    for(int i = 0; i < rez_r; i++) {
28        for(int j = 0; j < rez_c; j++) {
29            int max_val = mat[i*pool_size][j*pool_size];
30            for(int x = 0; x < pool_size; x++) {
31                for(int y = 0; y < pool_size; y++) {
32                    int val = mat[i*pool_size + x][j*pool_size + y];
33                    if(val > max_val) {
34                        max_val = val;
35                    }
36                rezultat[i][j] = max_val;
37            }
38        }
39        printf("\n");
40    }
41    return 0;
42 }
43 }
```

Output

```

Numar randuri: 2
Numar coloane: 2
Introduceti matricea:
13 15
17 19
Dimensiune fereastra pooling : 2
Matricea dupa max-pooling (2x2):
19
```

#Probleme pentru siruri de caractere:

Input

```
File  
(Ctrl+M)  
1 #include <stdio.h>  
2 #include <string.h>  
3 #define MAX 1005  
4 int main() {  
5     char str[MAX];  
6     char paranteze[MAX];  
7     int s = -1;  
8     int adv = 1;  
9     printf("Introdu sirul: ");  
10    fgets(str, 1000, stdin);  
11    str[strcspn(str, "\n")] = 0;  
12    for (int i = 0; str[i] != '\0'; i++) {  
13        if (str[i] == '(' || str[i] == '{' || str[i] == '[') {  
14            paranteze[++s] = str[i];  
15        } else if (str[i] == ')' || str[i] == '}' || str[i] == ']') {  
16            if (s == -1){  
17                adv = 0;  
18                break;  
19            }  
20            if ((str[i] == ')' && paranteze[s] != '(') || (str[i] == '}' && paranteze[s] != '{') ||  
21            (str[i] == ']' && paranteze[s] != '[')){  
22                adv = 0;  
23                break;  
24            }  
25            s--;}  
26        if (s != -1) {  
27            adv = 0;}  
28        if (adv) {  
29            printf("Parantezele sunt corecte.\n");} else {  
30            printf("Parantezele nu sunt corecte.\n");}  
31  
32    return 0;  
33 }
```

Output

```
Introdu sirul: {}[]  
Parantezele sunt corecte.  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

```
Introdu sirul: {0}[]  
Parantezele nu sunt corecte.  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir[100];
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     int lungime = 0;
10    while(sir[lungime] != '\0' && sir[lungime] != '\n') {
11        lungime++;
12    }
13
14    printf("Lungimea sirului: %d\n", lungime);
15
16    return 0;
17 }
```

Output

```
Introduceti un sir: Care este lungimea
Lungimea sirului: 18
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir1[100], sir2[100];
5
6     printf("Introduceti un sir: ");
7     fgets(sir1, sizeof(sir1), stdin);
8
9     int i = 0;
10    while(sir1[i] != '\0' && sir1[i] != '\n') {
11        i++;
12    }
13    sir1[i] = '\0';
14
15    i = 0;
16    while(sir1[i] != '\0') {
17        sir2[i] = sir1[i];
18        i++;
19    }
20    sir2[i] = '\0';
21
22    printf("Sirul original: %s\n", sir1);
23    printf("Sirul copiat: %s\n", sir2);
24
25    return 0;
26 }
```

Output

```
Introduceti un sir: Bine ai venit
sirul original: Bine ai venit
sirul copiat: Bine ai venit
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir1[100], sir2[100], rezultat[200];
5
6     printf("Introduceti primul sir: ");
7     fgets(sir1, sizeof(sir1), stdin);
8
9     printf("Introduceti al doilea sir: ");
10    fgets(sir2, sizeof(sir2), stdin);
11    |    int i = 0;
12    |    while(sir1[i] != '\0' && sir1[i] != '\n') {
13    |        i++;
14    |    }
15    |    sir1[i] = '\0';
16
17    i = 0;
18    while(sir2[i] != '\0' && sir2[i] != '\n') {
19        i++;
20    }
21    sir2[i] = '\0';
22    int j = 0;
23    i = 0;
24    while(sir1[i] != '\0') {
25        rezultat[j++] = sir1[i++];
26    }
27
28    i = 0;
29    while(sir2[i] != '\0') {
30        rezultat[j++] = sir2[i++];
31    }
32    rezultat[j] = '\0';
33    printf("Sirul concatenat: %s\n", rezultat);
34    return 0;
35 }
```

Output

```
Introduceti primul sir: Hello world!
Introduceti al doilea sir: Welcome!
Sirul concatenat: Hello world! Welcome!
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir1[100], sir2[100];
5
6     printf("Introduceti primul sir: ");
7     fgets(sir1, sizeof(sir1), stdin);
8
9     printf("Introduceti al doilea sir: ");
10    fgets(sir2, sizeof(sir2), stdin);
11    int i = 0;
12    while(sir1[i] != '\0' && sir1[i] != '\n') {
13        i++;
14    }
15    sir1[i] = '\0';
16
17    i = 0;
18    while(sir2[i] != '\0' && sir2[i] != '\n') {
19        i++;
20    }
21    sir2[i] = '\0';
22
23 // Comparare
24 i = 0;
25 int egale = 1;
26 while(sir1[i] != '\0' && sir2[i] != '\0') {
27     if(sir1[i] != sir2[i]) {
28         egale = 0;
29         break;
30     }
31     i++;
32 }
33 if(egale && sir1[i] == sir2[i]) {
34     printf("Sirurile sunt egale.\n");
35 } else {
36     printf("Sirurile NU sunt egale.\n");
37 }
38
39 return 0;
40 }
```

Output

```
Introduceti primul sir: Hello World
Introduceti al doilea sir: Hello World
Sirurile sunt egale.
```

Input

```
main.c
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9     int i = 0;
10    while(sir[i] != '\0' && sir[i] != '\n') {
11        i++;
12    }
13    sir[i] = '\0';
14    i = 0;
15    while(sir[i] != '\0') {
16        if(sir[i] >= 'a' && sir[i] <= 'z') {
17            sir[i] = sir[i] - 32;
18        }
19        i++;
20    }
21
22    printf("Sirul in majuscule: %s\n", sir);
23
24    return 0;
25 }
```

Output

```
Introduceti un sir: this is a testing code
Sirul in majuscule: THIS IS A TESTING CODE
```

Input

```
1 #include <stdio.h>
2
3 int main() {
4     char sir[100];
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8     |     int i = 0;
9     |     while(sir[i] != '\0' && sir[i] != '\n') {
10         |         i++;
11     }
12     |     sir[i] = '\0';
13     |     i = 0;
14     |     while(sir[i] != '\0') {
15         |         if(sir[i] >= 'A' && sir[i] <= 'Z') {
16             |             sir[i] = sir[i] + 32;
17         }
18         |         i++;
19     }
20
21     printf("Sirul in minuscule: %s\n", sir);
22
23     return 0;
24 }
```

Output

```
Introduceti un sir: THIS IS A TESTING CODE
Sirul in minuscule: this is a testing code
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir[100];
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     int i = 0;
10    while(sir[i] != '\0' && sir[i] != '\n') {
11        i++;
12    }
13    sir[i] = '\0';
14
15    i = 0;
16    while(sir[i] != '\0') {
17        if(sir[i] >= 'a' && sir[i] <= 'z') {
18            sir[i] = sir[i] - 32;
19        } else if(sir[i] >= 'A' && sir[i] <= 'Z') {
20            sir[i] = sir[i] + 32;
21        }
22        i++;
23    }
24
25    printf("Sirul cu litere comutate: %s\n", sir);
26
27    return 0;
28 }
```

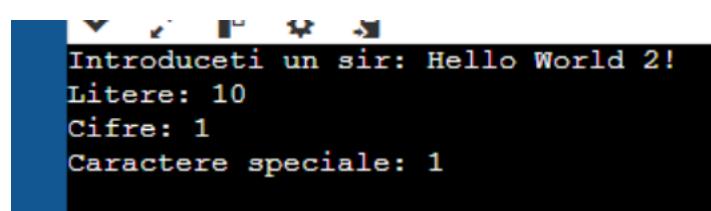
Output

```
Introduceti un sir: Welcome
Sirul cu litere comutate: wELCOME
```

Input

```
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int litere = 0, cifre = 0, speciale = 0;
11    int i = 0;
12
13    while(sir[i] != '\0' && sir[i] != '\n') {
14        if((sir[i] >= 'a' && sir[i] <= 'z') ||
15            (sir[i] >= 'A' && sir[i] <= 'Z')) {
16            litere++;
17        } else if(sir[i] >= '0' && sir[i] <= '9') {
18            cifre++;
19        } else if(sir[i] != ' ') {
20            speciale++;
21        }
22        i++;
23    }
24
25    printf("Litere: %d\n", litere);
26    printf("Cifre: %d\n", cifre);
27    printf("Caractere speciale: %d\n", speciale);
28
29    return 0;
30 }
```

Output



```
Introduceti un sir: Hello World 2!
Litere: 10
Cifre: 1
Caractere speciale: 1
```

Input

```
main.c
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int vocala = 0;
11    int i = 0;
12
13    while(sir[i] != '\0' && sir[i] != '\n') {
14        char c = tolower(sir[i]);
15        if(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
16            vocala++;
17        }
18        i++;
19    }
20
21    printf("Numar vocala: %d\n", vocala);
22
23    return 0;
24 }
```

Output

```
Introduceti un sir: This is a testing code
Numar vocala: 7
```

Input

```
main.c
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int consoane = 0;
11    int i = 0;
12
13    while(sir[i] != '\0' && sir[i] != '\n') {
14        char c = tolower(sir[i]);
15        if(c >= 'a' && c <= 'z') {
16            if(c != 'a' && c != 'e' && c != 'i' && c != 'o' && c != 'u') {
17                consoane++;
18            }
19        }
20        i++;
21    }
22
23    printf("Numar consoane: %d\n", consoane);
24
25    return 0;
26 }
```

Output

```
Introduceti un sir: This is a testing code
Numar consoane: 11
```

Input

```
main.c
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int cuvinte = 0;
11    int i = 0;
12
13    while(sir[i] != '\0' && sir[i] != '\n') {
14        if(sir[i] != ' ' && (sir[i+1] == ' ' || sir[i+1] == '\0' || sir[i+1] == '\n')) {
15            cuvinte++;
16        }
17        i++;
18    }
19
20    printf("Numar cuvinte: %d\n", cuvinte);
21
22    return 0;
23 }
```

Output

```
Introduceti un sir: this is a testing program
Numar cuvinte: 5
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir[100], invers[100];
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     int lungime = 0;
10    while(sir[lungime] != '\0' && sir[lungime] != '\n') {
11        lungime++;
12    }
13    sir[lungime] = '\0';
14
15    for(int i = 0; i < lungime; i++) {
16        invers[i] = sir[lungime - 1 - i];
17    }
18    invers[lungime] = '\0';
19
20    printf("Sirul original: %s\n", sir);
21    printf("Sirul inversat: %s\n", invers);
22
23    return 0;
24 }
```

Output

```
Introduceti un sir: Welcome
Sirul original: Welcome
Sirul inversat: emocleW
```

Input

```
main.c
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9     int lungime = 0;
10    while(sir[lungime] != '\0' && sir[lungime] != '\n') {
11        lungime++;
12    }
13    sir[lungime] = '\0';
14    int estePalindrom = 1;
15    for(int i = 0; i < lungime/2; i++) {
16        if(tolower(sir[i]) != tolower(sir[lungime - 1 - i])) {
17            estePalindrom = 0;
18            break;
19        }
20    }
21
22    if(estePalindrom) {
23        printf("Sirul este palindrom.\n");
24    } else {
25        printf("Sirul NU este palindrom.\n");
26    }
27
28    return 0;
29 }
```

Output

```
Introduceti un sir: Ana
Sirul este palindrom.
```

Input

```
main.c File
(Ctrl+M)
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     char cuvinte[50][50];
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9     sir[strcspn(sir, "\n")] = '\0';
10    int numarCuvinte = 0;
11    int indexCuvant = 0;
12
13    for(int i = 0; sir[i] != '\0'; i++) {
14        if(sir[i] == ' ') {
15            cuvinte[numarCuvinte][indexCuvant] = '\0';
16            numarCuvinte++;
17            indexCuvant = 0;
18        } else {
19            cuvinte[numarCuvinte][indexCuvant] = sir[i];
20            indexCuvant++;
21        }
22    }
23    cuvinte[numarCuvinte][indexCuvant] = '\0';
24    numarCuvinte++;
25    printf("Cuvintele in ordine inversa: ");
26    for(int i = numarCuvinte - 1; i >= 0; i--) {
27        printf("%s ", cuvinte[i]);
28    }
29    printf("\n");
30
31    return 0;
32 }
```

Output

```
Introduceti un sir: Welcome world
Cuvintele in ordine inversa: world Welcome
```

Input

```
main.c
1 #include <stdio.h>
2 int main() {
3     char sir[100], caracter;
4
5     printf("Introduceti un sir: ");
6     fgets(sir, sizeof(sir), stdin);
7
8     printf("Introduceti caracterul de cautat: ");
9     scanf("%c", &caracter);
10
11    int pozitie = -1;
12    int i = 0;
13
14    while(sir[i] != '\0' && sir[i] != '\n') {
15        if(sir[i] == caracter) {
16            pozitie = i;
17            break;
18        }
19        i++;
20    }
21
22    if(pozitie != -1) {
23        printf("Prima aparitie a caracterului '%c' este la pozitia: %d\n", caracter, pozitie);
24    } else {
25        printf("Caracterul '%c' nu a fost gasit in sir.\n", caracter);
26    }
27
28    return 0;
29 }
```

Output

```
Introduceti un sir: First Word
Introduceti caracterul de cautat: Word
Prima aparitie a caracterului 'W' este la pozitia: 6
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir[100], caracter;
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     printf("Introduceti caracterul de cautat: ");
10    scanf("%c", &caracter);
11
12    int pozitie = -1;
13    int i = 0;
14
15    while(sir[i] != '\0' && sir[i] != '\n') {
16        if(sir[i] == caracter) {
17            pozitie = i;
18        }
19        i++;
20    }
21
22    if(pozitie != -1) {
23        printf("Ultima aparitie a caracterului '%c' este la pozitia: %d\n", caracter, pozitie);
24    } else {
25        printf("Caracterul '%c' nu a fost gasit in sir.\n", caracter);
26    }
27
28    return 0;
29 }
```

Output

```
Introduceti un sir: Ultima litera este e
Introduceti caracterul de cautat: e
Ultima aparitie a caracterului 'e' este la pozitia: 19
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir[100], caracter;
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     printf("Introduceti caracterul de cautat: ");
10    scanf("%c", &caracter);
11
12    printf("Caracterul '%c' apare la pozitiile: ", caracter);
13    int gasit = 0;
14    int i = 0;
15
16    while(sir[i] != '\0' && sir[i] != '\n') {
17        if(sir[i] == caracter) {
18            printf("%d ", i);
19            gasit = 1;
20        }
21        i++;
22    }
23
24    if(!gasit) {
25        printf("nicio pozitie");
26    }
27    printf("\n");
28
29    return 0;
30 }
```

Output

```
Introduceti un sir: Welcome World!123
Introduceti caracterul de cautat: W
Caracterul 'W' apare la pozitiile: 0 8
```

Input

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char sir[100], caracter;
5
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     printf("Introduceti caracterul de numarat: ");
10    scanf("%c", &caracter);
11
12    int count = 0;
13    int i = 0;
14
15    while(sir[i] != '\0' && sir[i] != '\n') {
16        if(sir[i] == caracter) {
17            count++;
18        }
19        i++;
20    }
21
22    printf("Caracterul '%c' apare de %d ori.\n", caracter, count);
23
24    return 0;
25 }
```

Output

```
Introduceti un sir: Hello World
Introduceti caracterul de numarat: o
Caracterul 'o' apare de 2 ori.
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9     int lungime = strlen(sir);
10    if(sir[lungime-1] == '\n') {
11        sir[lungime-1] = '\0';
12        lungime--;
13    }
14    int frecventa[256] = {0};
15    int maxFrecventa = 0;
16    char caracterMax;
17    for(int i = 0; i < lungime; i++) {
18        if(sir[i] != ' ') {
19            frecventa[(int)sir[i]]++;
20            if(frecventa[(int)sir[i]] > maxFrecventa) {
21                maxFrecventa = frecventa[(int)sir[i]];
22                caracterMax = sir[i];
23            }
24        }
25    }
26    printf("Caracterul cu cea mai mare frecventa este '%c' (%d aparitii).\n",
27           caracterMax, maxFrecventa);
28
29    return 0;
30 }
```

Output

```
Introduceti un sir: Welcome program this is a testing program
Caracterul cu cea mai mare frecventa este 'r' (4 aparitii).
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3 #include <limits.h>
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8     int lungime = strlen(sir);
9     if(sir[lungime-1] == '\n') {
10         sir[lungime-1] = '\0';
11         lungime--;
12     }
13     int frecventa[256] = {0};
14     for(int i = 0; i < lungime; i++) {
15         if(sir[i] != ' ') {
16             frecventa[(int)sir[i]]++;
17         }
18     }
19     int minFrecventa = INT_MAX;
20     char caracterMin;
21
22     for(int i = 0; i < 256; i++) {
23         if(frecventa[i] > 0 && frecventa[i] < minFrecventa) {
24             minFrecventa = frecventa[i];
25             caracterMin = (char)i;
26         }
27     }
28     if(minFrecventa != INT_MAX) {
29         printf("Caracterul cu cea mai mica frecventa este '%c' (%d aparitii).\n",
30                caracterMin, minFrecventa);
31     } else {
32         printf("Sirul este gol sau contine doar spatii.\n");
33     }
34
35     return 0;
}
```

Output

```
Introduceti un sir: Welcome this is a testing program
Caracterul cu cea mai mica frecventa este 'W' (1 aparitii).
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     int lungime = strlen(sir);
10    if(sir[lungime-1] == '\n') {
11        sir[lungime-1] = '\0';
12        lungime--;
13    }
14
15    int frecventa[256] = {0};
16
17    for(int i = 0; i < lungime; i++) {
18        frecventa[(unsigned char)sir[i]]++;
19    }
20
21    printf("Frecventa caracterelor:\n");
22    for(int i = 0; i < 256; i++) {
23        if(frecventa[i] > 0) {
24            if(i == ' ') {
25                printf("Spatiu: %d\n", frecventa[i]);
26            } else if(i == '\t') {
27                printf("TAB: %d\n", frecventa[i]);
28            } else {
29                printf("%c: %d\n", i, frecventa[i]);
30            }
31        }
32    }
33
34    return 0;
35 }
```

Output

```
Introduceti un sir: este un sir de test
Frecventa caracterelor:
Spatiu: 4
'd': 1
'e': 4
'i': 1
'n': 1
'r': 1
's': 3
't': 3
'u': 1
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char caracter;
10    printf("Introduceti caracterul de eliminat: ");
11    scanf("%c", &caracter);
12
13    int lungime = strlen(sir);
14    if(sir[lungime-1] == '\n') {
15        sir[lungime-1] = '\0';
16        lungime--;
17    }
18
19    int pozitie = -1;
20    for(int i = 0; i < lungime; i++) {
21        if(sir[i] == caracter) {
22            pozitie = i;
23            break;
24        }
25    }
26
27    if(pozitie != -1) {
28        for(int i = pozitie; i < lungime; i++) {
29            sir[i] = sir[i+1];
30        }
31        printf("Sirul dupa eliminare: %s\n", sir);
32    } else {
33        printf("Caracterul nu a fost gasit in sir.\n");
34    }
35
36    return 0;
37 }
```

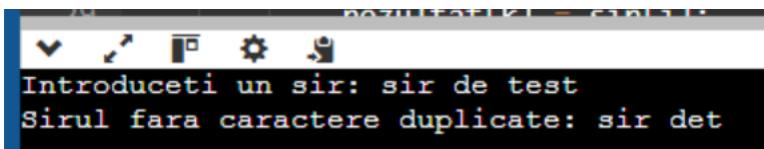
Output

```
Introduceti un sir: Sir de test
Introduceti caracterul de eliminat: de
Sirul dupa eliminare: Sir e test
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     int lungime = strlen(sir);
10    if(sir[lungime-1] == '\n') {
11        sir[lungime-1] = '\0';
12        lungime--;
13    }
14
15    char rezultat[100];
16    int k = 0;
17
18    for(int i = 0; i < lungime; i++) {
19        int gasit = 0;
20
21        for(int j = 0; j < i; j++) {
22            if(sir[i] == sir[j]) {
23                gasit = 1;
24                break;
25            }
26        }
27
28        if(!gasit) {
29            rezultat[k] = sir[i];
30            k++;
31        }
32    }
33    rezultat[k] = '\0';
34
35    printf("Sirul fara caractere duplicate: %s\n", rezultat);
36
37    return 0;
38 }
```

Output



A screenshot of a terminal window showing the output of the program. The window has a dark background with light-colored text. At the top, there are several small icons. Below them, the text "Introduceti un sir: sir de test" is displayed in white. At the bottom, the text "Sirul fara caractere duplicate: sir det" is displayed in white.

```
Introduceti un sir: sir de test
Sirul fara caractere duplicate: sir det
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char vechi, nou;
10    printf("Introduceti caracterul de inlocuit: ");
11    scanf(" %c", &vechi);
12    printf("Introduceti caracterul nou: ");
13    scanf(" %c", &nou);
14
15    int lungime = strlen(sir);
16    if(sir[lungime-1] == '\n') {
17        sir[lungime-1] = '\0';
18        lungime--;
19    }
20
21    int schimbat = 0;
22    for(int i = 0; i < lungime; i++) {
23        if(sir[i] == vechi && !schimbat) {
24            sir[i] = nou;
25            schimbat = 1;
26        }
27    }
28
29    if(schimbat) {
30        printf("Sirul dupa inlocuire: %s\n", sir);
31    } else {
32        printf("Caracterul nu a fost gasit in sir.\n");
33    }
34
35    return 0;
36 }
```

Output

```
Introduceti un sir: Welcome
Introduceti caracterul de inlocuit: W e
Introduceti caracterul nou: Sirul dupa inlocuire: eelcome
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char vechi, nou;
10    printf("Introduceti caracterul de inlocuit: ");
11    scanf(" %c", &vechi);
12    printf("Introduceti caracterul nou: ");
13    scanf(" %c", &nou);
14
15    int lungime = strlen(sir);
16    if(sir[lungime-1] == '\n') {
17        sir[lungime-1] = '\0';
18        lungime--;
19    }
20
21    int pozitie = -1;
22    for(int i = 0; i < lungime; i++) {
23        if(sir[i] == vechi) {
24            pozitie = i;
25        }
26    }
27
28    if(pozitie != -1) {
29        sir[pozitie] = nou;
30        printf("Sirul dupa inlocuire: %s\n", sir);
31    } else {
32        printf("Caracterul nu a fost gasit in sir.\n");
33    }
34
35    return 0;
36 }
```

Output

```
Introduceti un sir: Welcome Home
Introduceti caracterul de inlocuit: H z
Introduceti caracterul nou: Sirul dupa inlocuire: Welcome zome
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de cautat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12
13    int lungime_sir = strlen(sir);
14    if(sir[lungime_sir-1] == '\n') {
15        sir[lungime_sir-1] = '\0';
16    }
17
18    int lungime_cuvant = strlen(cuvant);
19    if(cuvant[lungime_cuvant-1] == '\n') {
20        cuvant[lungime_cuvant-1] = '\0';
21        lungime_cuvant--;
22    }
23
24    char *pozitie = strstr(sir, cuvant);
25
26    if(pozitie != NULL) {
27        int index = pozitie - sir;
28        printf("Prima aparitie a cuvantului '%s' este la pozitia: %d\n", cuvant, index);
29    } else {
30        printf("Cuvantul '%s' nu a fost gasit in sir.\n", cuvant);
31    }
32
33    return 0;
34 }
```

Output

```
Introduceti un sir: este un sir de test
Introduceti cuvantul de cautat: un
Prima aparitie a cuvantului 'un' este la pozitia: 5
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de cautat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12
13    int lungime_sir = strlen(sir);
14    if(sir[lungime_sir-1] == '\n') {
15        sir[lungime_sir-1] = '\0';
16    }
17    int lungime_cuvant = strlen(cuvant);
18    if(cuvant[lungime_cuvant-1] == '\n') {
19        cuvant[lungime_cuvant-1] = '\0';
20        lungime_cuvant--;
21    }
22    char *ultima_pozitie = NULL;
23    char *current = sir;
24    while((current = strstr(current, cuvant)) != NULL) {
25        ultima_pozitie = current;
26        current++;
27    }
28
29    if(ultima_pozitie != NULL) {
30        int index = ultima_pozitie - sir;
31        printf("Ultima aparitie a cuvantului '%s' este la pozitia: %d\n", cuvant, index);
32    } else {
33        printf("Cuvantul '%s' nu a fost gasit in sir.\n", cuvant);
34    }
35
36    return 0;
37 }
```

Output

```
Introduceti un sir: este un sir de test
Introduceti cuvantul de cautat: de
Ultima aparitie a cuvantului 'de' este la pozitia: 12
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de cautat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12    int lungime_sir = strlen(sir);
13    if(sir[lungime_sir-1] == '\n') {
14        sir[lungime_sir-1] = '\0';
15    }
16    int lungime_cuvant = strlen(cuvant);
17    if(cuvant[lungime_cuvant-1] == '\n') {
18        cuvant[lungime_cuvant-1] = '\0';
19        lungime_cuvant--;
20    }
21    printf("Cuvantul '%s' apare la pozitiile: ", cuvant);
22    char *current = sir;
23    int gasit = 0;
24    while((current = strstr(current, cuvant)) != NULL) {
25        int index = current - sir;
26        printf("%d ", index);
27        gasit = 1;
28        current++;
29    }
30    if(!gasit) {
31        printf("nicio pozitie");
32    }
33    printf("\n");
34    return 0;
35 }
```

Output

```
Introduceti un sir: test acesta este un sir de test
Introduceti cuvantul de cautat: test
Cuvantul 'test' apare la pozitiile: 0 27
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de numarat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12    int lungime_sir = strlen(sir);
13    if(sir[lungime_sir-1] == '\n') {
14        sir[lungime_sir-1] = '\0';
15    }
16
17    int lungime_cuvant = strlen(cuvant);
18    if(cuvant[lungime_cuvant-1] == '\n') {
19        cuvant[lungime_cuvant-1] = '\0';
20        lungime_cuvant--;
21    }
22
23    int count = 0;
24    char *curent = sir;
25
26    while((curent = strstr(curent, cuvant)) != NULL) {
27        count++;
28        curent += lungime_cuvant;
29    }
30
31    printf("Cuvantul '%s' apare de %d ori.\n", cuvant, count);
32
33    return 0;
34 }
```

Output

```
Introduceti un sir: hello world!
Introduceti cuvantul de numarat: l
Cuvantul 'l' apare de 3 ori.
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de eliminat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12
13    int lungime_sir = strlen(sir);
14    if(sir[lungime_sir-1] == '\n') {
15        sir[lungime_sir-1] = '\0';
16        lungime_sir--;
17    }
18
19    int lungime_cuvant = strlen(cuvant);
20    if(cuvant[lungime_cuvant-1] == '\n') {
21        cuvant[lungime_cuvant-1] = '\0';
22        lungime_cuvant--;
23    }
24
25    char *pozitie = strstr(sir, cuvant);
26
27    if(pozitie != NULL) {
28        int index = pozitie - sir;
29
30        for(int i = index; i <= lungime_sir - lungime_cuvant; i++) {
31            sir[i] = sir[i + lungime_cuvant];
32        }
33
34        printf("Sirul dupa eliminare: %s\n", sir);
35    } else {
36        printf("Cuvantul nu a fost gasit in sir.\n");
37    }
38    return 0;
39 }
```

Output

```
Introduceti un sir: hello world
Introduceti cuvantul de eliminat: world
Sirul dupa eliminare: hello
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de eliminat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12
13    int lungime_sir = strlen(sir);
14    if(sir[lungime_sir-1] == '\n') {
15        sir[lungime_sir-1] = '\0';
16        lungime_sir--;
17    }
18    int lungime_cuvant = strlen(cuvant);
19    if(cuvant[lungime_cuvant-1] == '\n') {
20        cuvant[lungime_cuvant-1] = '\0';
21        lungime_cuvant--;
22    }
23    char *ultima_pozitie = NULL;
24    char *current = sir;
25
26    while((current = strstr(current, cuvant)) != NULL) {
27        ultima_pozitie = current;
28        current++;
29    }
30    if(ultima_pozitie != NULL) {
31        int index = ultima_pozitie - sir;
32
33        for(int i = index; i <= lungime_sir - lungime_cuvant; i++) {
34            sir[i] = sir[i + lungime_cuvant];
35        }
36
37        printf("Sirul dupa eliminare: %s\n", sir);
38    } else {
39        printf("Cuvantul nu a fost gasit in sir.\n");
40    }
41
42    return 0;
43 }
```

Output

```
Introduceti un sir: este un sir de de test
Introduceti cuvantul de eliminat: de
Sirul dupa eliminare: este un sir de  test
```

Input

```

main.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char sir[100];
6     printf("Introduceti un sir: ");
7     fgets(sir, sizeof(sir), stdin);
8
9     char cuvant[50];
10    printf("Introduceti cuvantul de eliminat: ");
11    fgets(cuvant, sizeof(cuvant), stdin);
12
13    int lungime_sir = strlen(sir);
14    if(sir[lungime_sir-1] == '\n') {
15        sir[lungime_sir-1] = '\0';
16        lungime_sir--;
17    }
18
19    int lungime_cuvant = strlen(cuvant);
20    if(cuvant[lungime_cuvant-1] == '\n') {
21        cuvant[lungime_cuvant-1] = '\0';
22        lungime_cuvant--;
23    }
24
25    char rezultat[200] = "";
26    char *cuvant_curent = strtok(sir, " ");
27    int prima = 1;
28
29    while(cuvant_curent != NULL) {
30        if(strcmp(cuvant_curent, cuvant) != 0) {
31            if(!prima) {
32                strcat(rezultat, " ");
33            }
34            strcat(rezultat, cuvant_curent);
35            prima = 0;
36        }
37        cuvant_curent = strtok(NULL, " ");
38    }
39
40    printf("Sirul dupa eliminare: %s\n", rezultat);
41
42    return 0;
43 }

```

Output

```

Introduceti un sir: acesta este un sir de test, sir rar
Introduceti cuvantul de eliminat: sir
Sirul dupa eliminare: acesta este un de test, rar

```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4
5 int main() {
6     char sir[100];
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int lungime = strlen(sir);
11    if(sir[lungime-1] == '\n') {
12        sir[lungime-1] = '\0';
13        lungime--;
14    }
15
16    int start = 0;
17    while(isisspace(sir[start])) {
18        start++;
19    }
20    for(int i = 0; i < lungime - start; i++) {
21        sir[i] = sir[i + start];
22    }
23    sir[lungime - start] = '\0';
24
25    printf("Sirul fara spatii la inceput: '%s'\n", sir);
26
27    return 0;
28 }
```

Output

```
Introduceti un sir: este un sir de test
Sirul fara spatii la inceput: 'este un sir de test'
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4
5 int main() {
6     char sir[100];
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int lungime = strlen(sir);
11    if(sir[lungime-1] == '\n') {
12        sir[lungime-1] = '\0';
13        lungime--;
14    }
15
16    int end = lungime - 1;
17    while(end >= 0 && isspace(sir[end])) {
18        end--;
19    }
20
21    sir[end + 1] = '\0';
22
23    printf("Sirul fara spatii la sfarsit: '%s'\n", sir);
24
25    return 0;
26 }
```

Output

```
Introduceti un sir: este un sir de test
Sirul fara spatii la sfarsit: 'este un sir de test'
```

Input

```
main.c
2 #include <string.h>
3 #include <ctype.h>
4
5 int main() {
6     char sir[100];
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9     int lungime = strlen(sir);
10    if(sir[lungime-1] == '\n') {
11        sir[lungime-1] = '\0';
12        lungime--;
13    }
14    int start = 0;
15    while(ispace(sir[start])) {
16        start++;
17    }
18    int end = lungime - 1;
19    while(end >= start && ispace(sir[end])) {
20        end--;
21    }
22    char rezultat[100];
23    int j = 0;
24    for(int i = start; i <= end; i++) {
25        rezultat[j] = sir[i];
26        j++;
27    }
28    rezultat[j] = '\0';
29
30    printf("Sirul dupa trimitere: '%s'\n", rezultat);
31
32    return 0;
33 }
```

Output

```
Introduceti un sir: este un sir de test
Sirul dupa trimitere: 'este un sir de test'
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4
5 int main() {
6     char sir[100];
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9     int lungime = strlen(sir);
10    if(sir[lungime-1] == '\n') {
11        sir[lungime-1] = '\0';
12        lungime--;
13    }
14    char litere[100];
15    int k = 0;
16    for(int i = 0; i < lungime; i++) {
17        if(isalpha(sir[i])) {
18            litere[k] = sir[i];
19            k++;
20        }
21    }
22    litere[k] = '\0';
23    for(int i = 0; i < k-1; i++) {
24        for(int j = 0; j < k-i-1; j++) {
25            if(tolower(litere[j]) > tolower(litere[j+1])) {
26                char temp = litere[j];
27                litere[j] = litere[j+1];
28                litere[j+1] = temp;
29            }
30        }
31    }
32    k = 0;
33    for(int i = 0; i < lungime; i++) {
34        if(isalpha(sir[i])) {
35            sir[i] = litere[k];
36            k++;
37        }
38    }
39    printf("Sirul cu litere sortate: %s\n", sir);
40
41    return 0;
42 }
```

Output

```
Introduceti un sir: Sir de test
Sirul cu litere sortate: dee ir Stt
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4
5 int main() {
6     char sir[100];
7     printf("Introduceti un sir: ");
8     fgets(sir, sizeof(sir), stdin);
9
10    int lungime = strlen(sir);
11    if(sir[lungime-1] == '\n') {
12        sir[lungime-1] = '\0';
13        lungime--;
14    }
15    char cuvinte[50][50];
16    int numar_cuvinte = 0;
17    char *cuvant = strtok(sir, " ");
18
19    while(cuvant != NULL && numar_cuvinte < 50) {
20        strcpy(cuvinte[numar_cuvinte], cuvant);
21        numar_cuvinte++;
22        cuvant = strtok(NULL, " ");
23    }
24    for(int i = 0; i < numar_cuvinte-1; i++) {
25        for(int j = 0; j < numar_cuvinte-i-1; j++) {
26            if(strcasecmp(cuvinte[j], cuvinte[j+1]) > 0) {
27                char temp[50];
28                strcpy(temp, cuvinte[j]);
29                strcpy(cuvinte[j], cuvinte[j+1]);
30                strcpy(cuvinte[j+1], temp);
31            }
32        }
33    }
34    printf("Cuvintele sortate: ");
35    for(int i = 0; i < numar_cuvinte; i++) {
36        printf("%s ", cuvinte[i]);
37    }
38    printf("\n");
39
40    return 0;
41 }
```

Output

```
Introduceti un sir: sir de test
Cuvintele sortate: de sir test
```

Input

Output

```
Introduceti o expresie matematica: (2*5)+7-2
Expresie matematica corecta.
```

Input

```
main.c
1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4 int main() {
5     char expresie[100];
6     printf("Introduceti o expresie matematica: ");
7     fgets(expresie, sizeof(expresie), stdin);
8     int lungime = strlen(expresie);
9     if(expresie[lungime-1] == '\n') {
10         expresie[lungime-1] = '\0';
11         lungime--;
12     }
13     int paranteze = 0;
14     int corecta = 1;
15     for(int i = 0; i < lungime; i++) {
16         char c = expresie[i];
17         if(!isdigit(c) &&
18             c != '+' && c != '-' && c != '*' && c != '/' &&
19             c != '(' && c != ')' && c != ' ') {
20             corecta = 0;
21             break;
22         }
23         if(corecta) { for(int i = 0; i < lungime; i++) {
24             if(expresie[i] == '(') {
25                 paranteze++;
26             } else if(expresie[i] == ')') {
27                 paranteze--;
28                 if(paranteze < 0) {
29                     corecta = 0;
30                     break;
31                 }
32                 for(int i = 0; i < lungime-1; i++) {
33                     char c1 = expresie[i];
34                     char c2 = expresie[i+1];
35                     if((c1 == '+' || c1 == '-' || c1 == '*' || c1 == '/') &&
36                         (c2 == '+' || c2 == '-' || c2 == '*' || c2 == '/')) {
37                         corecta = 0;
38                         break;
39                     }
40                     if(i == 0 && (c1 == '*' || c1 == '/' || c1 == ')')) {
41                         corecta = 0;
42                         break;
43                     }
44                     if(ultim == '+' || ultim == '-' || ultim == '*' || ultim == '/' || ultim == '(') {
45                         corecta = 0;
46                     }
47                     if(corecta) {
48                         printf("Expresie matematica corecta.\n");
49                     } else {
50                         printf("Expresie matematica gresita.\n");
51                     }
52                 }
53             }
54         }
55     }
56     return 0;
57 }
```

Output

```
Introduceti o expresie matematica: (2*5)+7-2
Expresie matematica corecta.
```

