F I R S T P R O J E C T O B J E C T I V E

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


int iterativesum(int, int);
int recursivesum(int, int);

int main()

{    /* Project's First Subject/Question. */


iterativesum(num, cycle){

recursivesum(num, cycle){
```

As the picture refers to the code was formed out of three major parts.

1. *Interface (Main Function).*
2. *Iterative Function.*
3. *Recursive Function.*

1-INTERFACE

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


int iterativesum(int, int);
int recursivesum(int, int);

int main()

{    /* Project's First Subject/Question. */

int num;
int cycle;
int counter;
int forrecursive;
    printf("Please set two integer for summation \n \n first integer : ");
            scanf("%d", &num);
        printf("\n Second integer : ");
            scanf("%d", &cycle);
                printf("\n  Iterative result is : %d\n ", iterativesum(num, cycle));
    printf("\n \n");
                  printf("  Recursive result is : %d\n\n", recursivesum(num, cycle));
}
```

1. To begin the process requests two integers.
2. These two integers were saved to the memory.
3. Then called out required functions for the result.

Output:

```
Please set two integer for summation

 first integer : 5

 Second integer : 2
```

As you can see, everything is clear for an user-friendly interface.

2-ITERATIVE FUNCTION

```
iterativesum(num, cycle){
int i1;
int i2;
int num1;
num1=num-1;
    for(i2=1; i2<cycle; i2++){
        for(i1=1; i1<=num1; i1++){
            num=num+i1;
            }
        num1=num-1;
    }
num1=num1+1;

return num1;
```

i2 has using for cycle loop and i1 has using for summation Loop

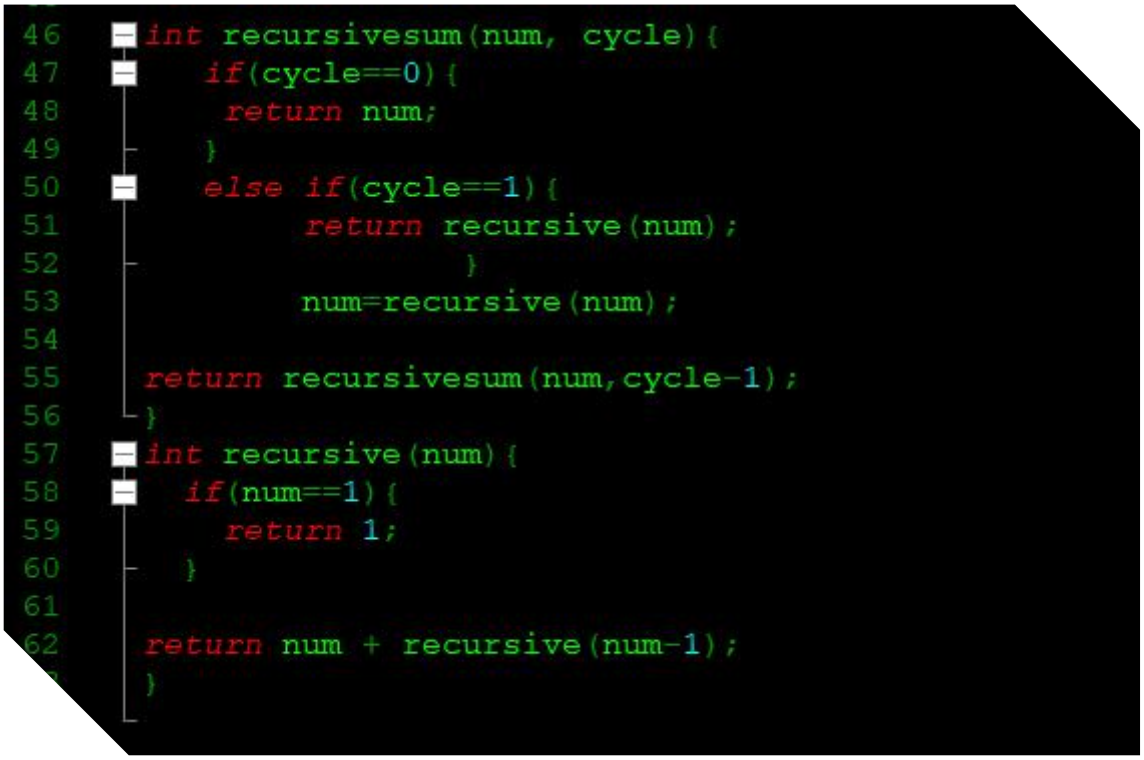
Output:

```
Please set two integer for summation
 first integer : 5
 Second integer : 2
  Iterative result is : 15
```

As you can see,

it's works without any issue.

3-RECURSIVE FUNCTION



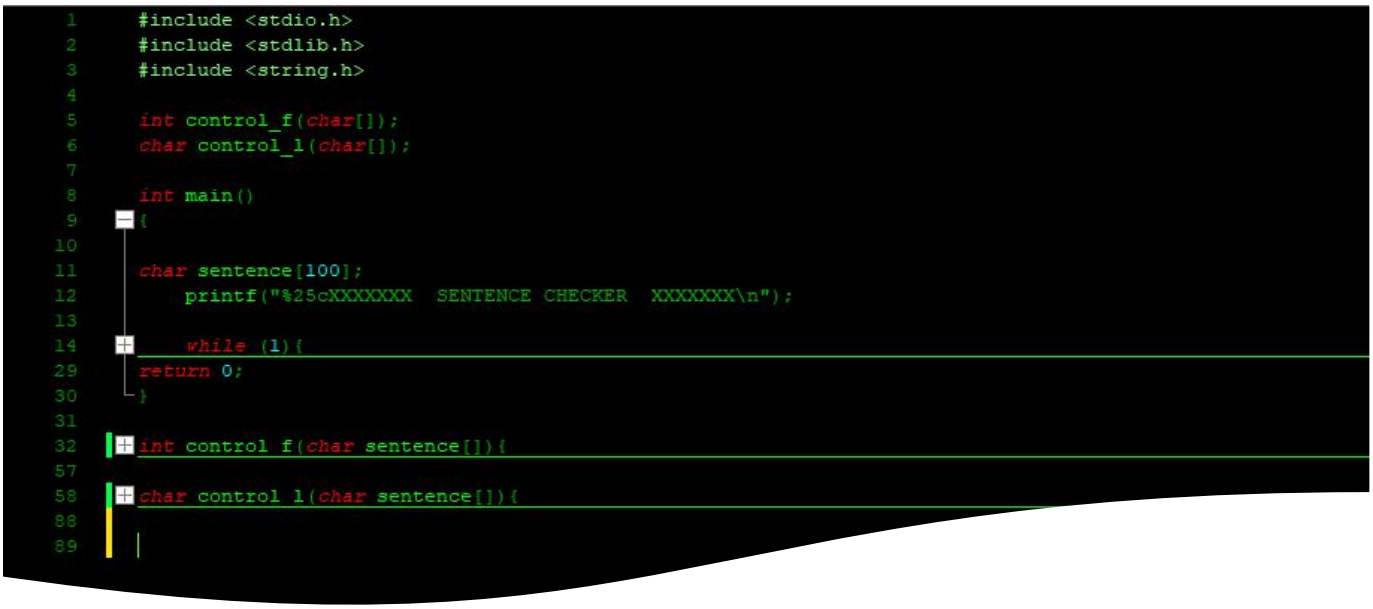
```
46  int recursivesum(num, cycle){
47      if(cycle==0){
48       return num;
49      }
50      else if(cycle==1){
51           return recursive(num);
52                    }
53           num=recursive(num);
54
55   return recursivesum(num,cycle-1);
56  }
57  int recursive(num){
58    if(num==1){
59      return 1;
60    }
61
62   return num + recursive(num-1);
   }
```

I use **if** and **else if** command for control

mechanism.

And define some counters for loops.

S E C O N D P R O J E C T O B J E C T I V E



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

int control_f(char[]);
char control_l(char[]);

int main()
{

char sentence[100];
    printf("%25cXXXXXXX  SENTENCE CHECKER  XXXXXXX\n");

    while (1){
return 0;
}

int control_f(char sentence[]){

char control_l(char sentence[]){

 |
```

As the picture refers to the code was formed out of three major parts.

1. *Interface .*
2. *Fully or Not.*
3. *Most Repeated Letter.*

1-INTERFACE

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

int control_f(char[]);
char control_l(char[]);

int main()
{

char sentence[100];
    printf("%25cXXXXXXX  SENTENCE CHECKER  XXXXXXX\n");

    while (1){
    printf("\n \nPlease enter a sentence : ");
    gets(sentence);


if(control_f(strlwr(sentence))==1){
        printf("\n"); puts(sentence);
        printf("is a fully sentence\n");
}
else{
        printf("\n  ----> "); puts(sentence);
        printf(" is not a fully sentence\n");
}
    printf("\nmost repeated letter is : %c\n",control_l(sentence));
    }
return 0;
}
```

1. To begin the process requests a sentence.

2. Sentence is save as array using “gets” command .

3. Then called out required functions for determine whether the sentence is fully and find the most repeated letter.

Output:

```
                        XXXXXXX  SENTENCE CHECKER  XXXXXXX

Please enter a sentence : is this sentence fully? and what is the most repeated letter in this sentence?
```

2-FULLY OR NOT

```
31
32  int control_f(char sentence[]){
33
34      char alphabet[]={'a','b','c','d',
35                       'e','f','g','h',
36                       'i','j','k','l',
37                       'm','n','o','p',
38                       'q','r','s','t',
39                       'u','v','w','x',
40                       'y','z'};
41  int a=0, b=0;
42  int c=0;
43      while(sentence[a]!='\0'){
44          for(b=0; b<26; b++){
45              if(sentence[a]==alphabet[b]){
46  alphabet[b]='-';
47  c++;
48              }
49          }
50  a++;
51      }
52      if(c==26){
53  return 1;
54      }
    return 0;
}
```

First I define the alphabet with an array.

Than define some counters for calculations and control mechanisms.

I use **for** and **if** functions for check alphabet .

Output:

```
                    XXXXXXX  SENTENCE CHECKER  XXXXXXX


Please enter a sentence : is this sentence fully? and what is the most repeated letter in this sentence?

  ----> is this sentence fully? and what is the most repeated letter in this sentence?
 is not a fully sentence

most repeated letter is : e


Please enter a sentence : The quick brown fox jumps over the lazy dog

the quick brown fox jumps over the lazy dog
is a fully sentence

most repeated letter is : o


Please enter a sentence :
```

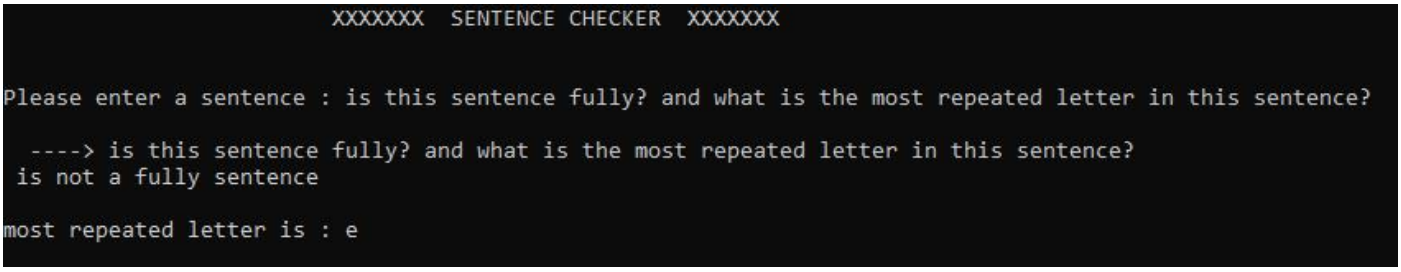
3-MOST REPEATED LETTER

First I define alphabet with an array.

Than define some counters and arrays for calculations and control mechanisms.

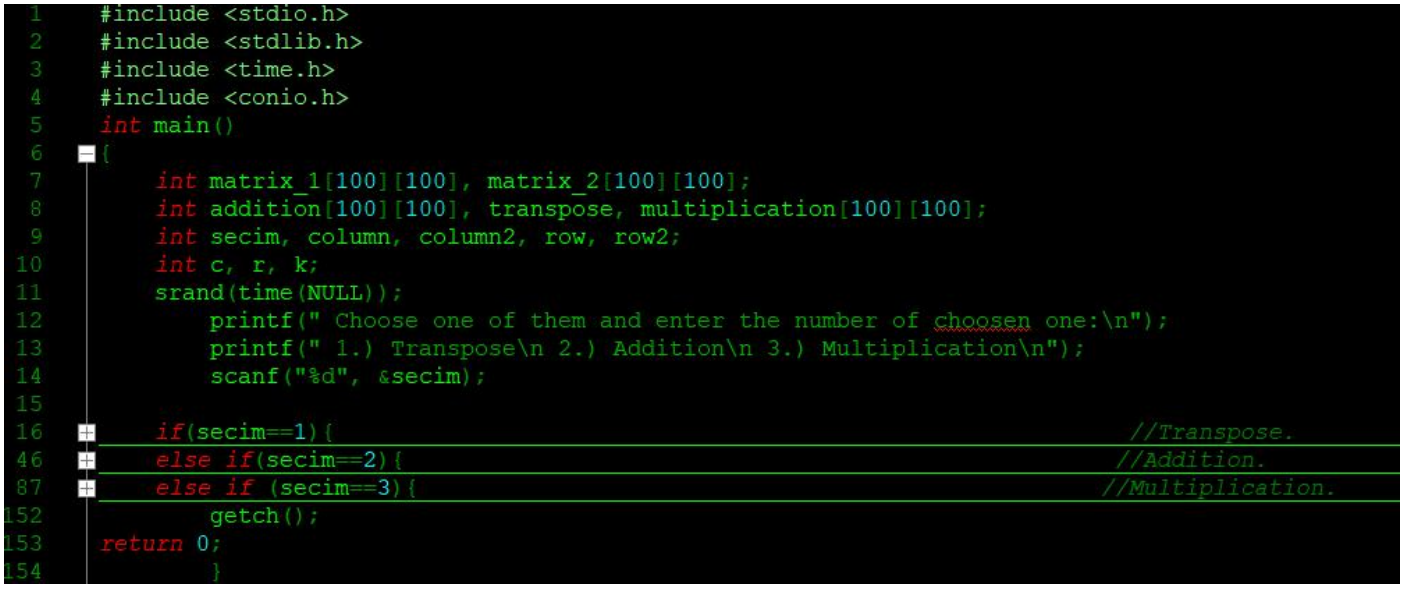
I use **for** and **if** functions for check all letters.

Output:



As you can see, it's works without any issue.

T H I R D P R O J E C T O B J E C T I V E



```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
int main()
{
    int matrix_1[100][100], matrix_2[100][100];
    int addition[100][100], transpose, multiplication[100][100];
    int secim, column, column2, row, row2;
    int c, r, k;
    srand(time(NULL));
        printf(" Choose one of them and enter the number of choosen one:\n");
        printf(" 1.) Transpose\n 2.) Addition\n 3.) Multiplication\n");
        scanf("%d", &secim);

    if(secim==1){                                                    //Transpose.
    else if(secim==2){                                               //Addition.
    else if (secim==3){                                              //Multiplication.
        getch();
return 0;
        }
```

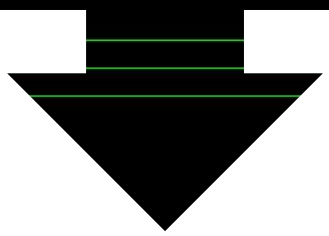
As the picture refers to the code was formed out of four major parts.

1. *Interface .*
2. *Transpose.*
3. *Addition.*
4. *Multiplication.*

1-INTERFACE

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
int main()
{
    int matrix_1[100][100], matrix_2[100][100];
    int addition[100][100], transpose, multiplication[100][100];
    int secim, column, column2, row, row2;
    int c, r, k;
    srand(time(NULL));
        printf(" Choose one of them and enter the number of choosen one:\n");
        printf(" 1.) Transpose\n 2.) Addition\n 3.) Multiplication\n");
        scanf("%d", &secim);

```



1. To begin the process requests a choosing operation.

2. Then called out required questions for the function.

Output:

```
Choose one of them and enter the number of choosen one:
1.) Transpose
2.) Addition
3.) Multiplication
1
```

As you can see everything is clear for an *user-friendly* interface.

2-TRANSPOSE

```
    int c, r, c_2;
    srand(time(NULL));
        printf("Matrix ile hangi islemi yapmak istiyorsun?\n");
        printf(" Choose one of them and enter the number of choosen one:\n");
        printf(" 1.) Transpose\n 2.) Addition\n 3.) Multiplication\n");
        scanf("%d", &secim);


    if(secim==1){                                                    //Transpose.
        printf("\n\nPlease enter dimensions what you want");
            printf("\nrow: ");      scanf("%d",&row);
            printf("\ncolumn: ");   scanf("%d",&column);

    for(c=0; c<column; c++){                                         //Random matris olusturma islemi.
        for(r=0; r<row; r++){
            matrix_1[r][c] = rand() % 100 ;
        }
    }

printf("\n matrix1 : \n");                                          //Matrisi gosteren kod.

    for(c=0; c<column; c++){
        for(r=0; r<row; r++)
            printf("%d\t",matrix_1[r][c]);
        printf("\n\n");
    }
        // i = r // j = c //

              printf("\n Transpose : \n")   ;                       //Matrisin Transpozesi.
  for (r = 0; r < row; r++)
    {
        for (c = 0; c<column; c++)                                   //sonucu gosteren kod.
        {
            printf("%4d", matrix_1[r][c]);
        }
        printf("\n\n\n");
    }

}
    else
    else
return 0;
```

if(secim==1) *shows that, if user choose first function-transpose- use then this algorithm.*

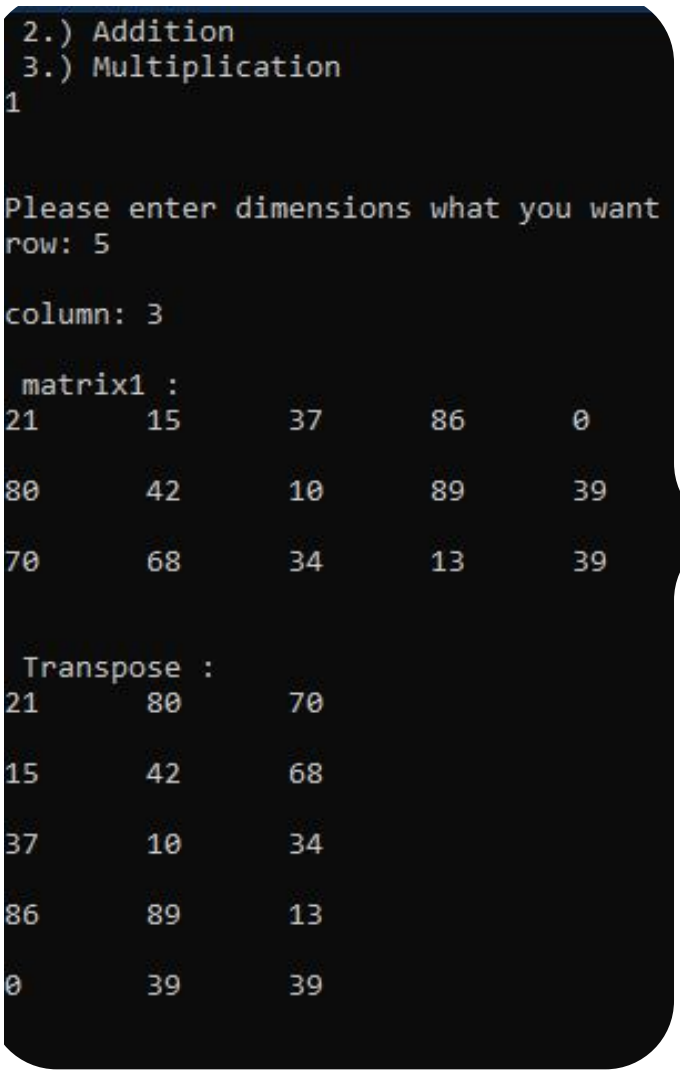
Printf and scanf functions these are requests to user for matrix's dimensions.

And save the data from user about row and column.

First and Second FOR Loops Are using for generate random matrix values with dimensions from user.

Last two FOR Loops Are using for printing generated matrix on screen.

2.1-TRANSPOSE OUTPUT



1. Choosing transpose function with write “1”.

2. Entering matrix’s dimensions.

3. Program prints “matrix1”.

4. Program prints “matrix1”s “Transpose”.

3-ADDITION

```
}
    else if(secim==2){                                          //Addition.

    printf("\n\nPlease enter first matrix's dimensions what you want");
            printf("\nrow: ");       scanf("%d",&row);
            printf("\ncolumn: ");    scanf("%d",&column);

    for(r=0; r<row; r++){                                       //Random 1.matris olusturma islemi.
        for(c=0; c<column; c++){
            matrix_1[r][c] = rand() % 100 ;
            }
        }
    for(r=0; r<row; r++){                                       //Random 1.matris olusturma islemi.
        for(c=0; c<column; c++){
            matrix_2[r][c] = rand() % 100 ;
            }
        }

        printf("\n matrix1 : \n");                               //1. Matrisi gosteren kod.
    for(c=0; c<column; c++){
        for(r=0; r<row; r++)
            printf("%d\t",matrix_1[r][c]);
        printf("\n\n");
    }
        printf("\n matrix2 : \n");                               //2. Matrisi gosteren kod.
    for(c=0; c<column; c++){
          for(r=0; r<row; r++)
            printf("%d\t",matrix_2[r][c]);
        printf("\n\n");
        }
 printf("\n ADDITION OF TWO MATRIX :\n");                       //Toplama isleminin yapilmasi.

 for(c=0; c<column; c++){
          for(r=0; r<row; r++)
                addition[r][c]=matrix_1[r][c]+matrix_2[r][c];
        }

for(c=0; c<column; c++){                                        //Sonucun gosterilmesi.
          for(r=0; r<row; r++)
            printf("%d\t",addition[r][c]);
        printf("\n");
    }
}
```

1.if(secim==2) *shows that, if user choose second function-Addition- then use this algorithm.*

2.printf and scanf functions these are requests to user for matrices's dimensions.

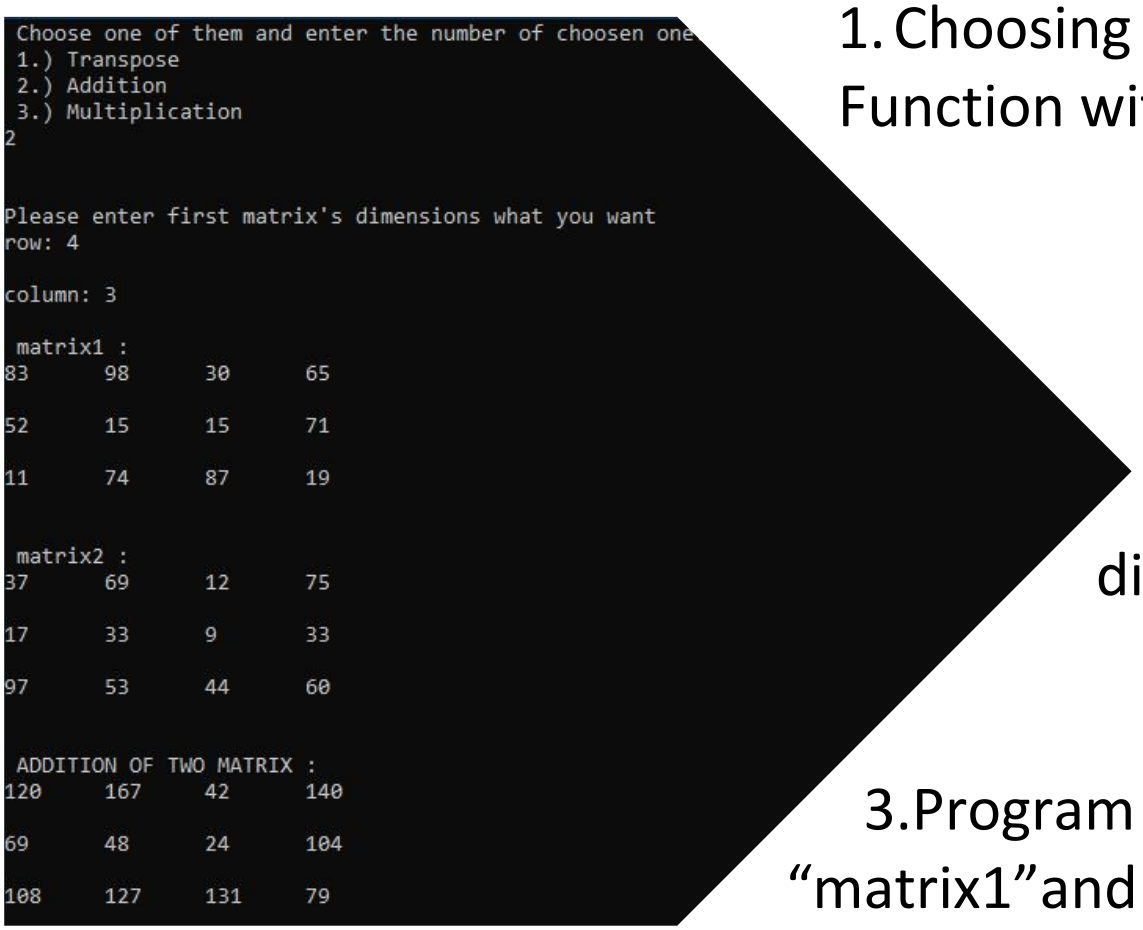
And save the data from user about rows and columns.

3.Four FOR Loops Are using for generate random matrices values with dimensions from user.

4.Last two FOR Loops Are using for printing generated matrices on screen.

5.Next two FOR Loops Are using for calculate the Addition

3.1-ADDITION OUTPUT



1. Choosing Addition Function with write ‘2’.

2.Entering matrices’s dimensions.

3.Program prints “matrix1”and “matrix2”.

4.Program prints “matrix1” + “matrix2”.

4-MULTIPLICATION

```
}
    else if (secim==3){                                          //Multiplication.

    printf("\n\nPlease enter first matrix's dimensions what you want"); //ilk matris boyutlarinin alinmasi.
            printf("\nrow: ");       scanf("%d",&row);
            printf("\ncolumn: ");    scanf("%d",&column);

        for(r=0; r<row; r++){
        for(c=0; c<column; c++){
            matrix_1[r][c] = rand() % 100 ;
            }
        }
    printf("\n\nPlease enter second matrix's dimensions what you want");//ikinci matris boyutlarinin alinmasi.
           printf("\nrow: ");       scanf("%d",&row2);
           printf("\ncolumn: ");    scanf("%d",&column2);

        for(r=0; r<row2; r++){
        for(c=0; c<column2; c++){
            matrix_2[r][c] = rand() % 100 ;
            }
        }
             printf("\n matrix1 : \n");                           //ilk matrisi random olusturma.

    for(c=0; c<column; c++){
        for(r=0; r<row; r++)
            printf("%d\t",matrix_1[r][c]);
        printf("\n\n");
    }
        printf("\n matrix2 : \n");                               //ikinci matrisi random olusturma.
          for(c=0; c<column2; c++){
          for(r=0; r<row2; r++)
            printf("%d\t",matrix_2[r][c]);
        printf("\n\n");
        }
    }
return 0;
```

1.if(secim==3) *shows that, if user choose second function-Multiplication- then use this algorithm.*

2.printf and scanf functions these are requests to user for matrices's dimensions.

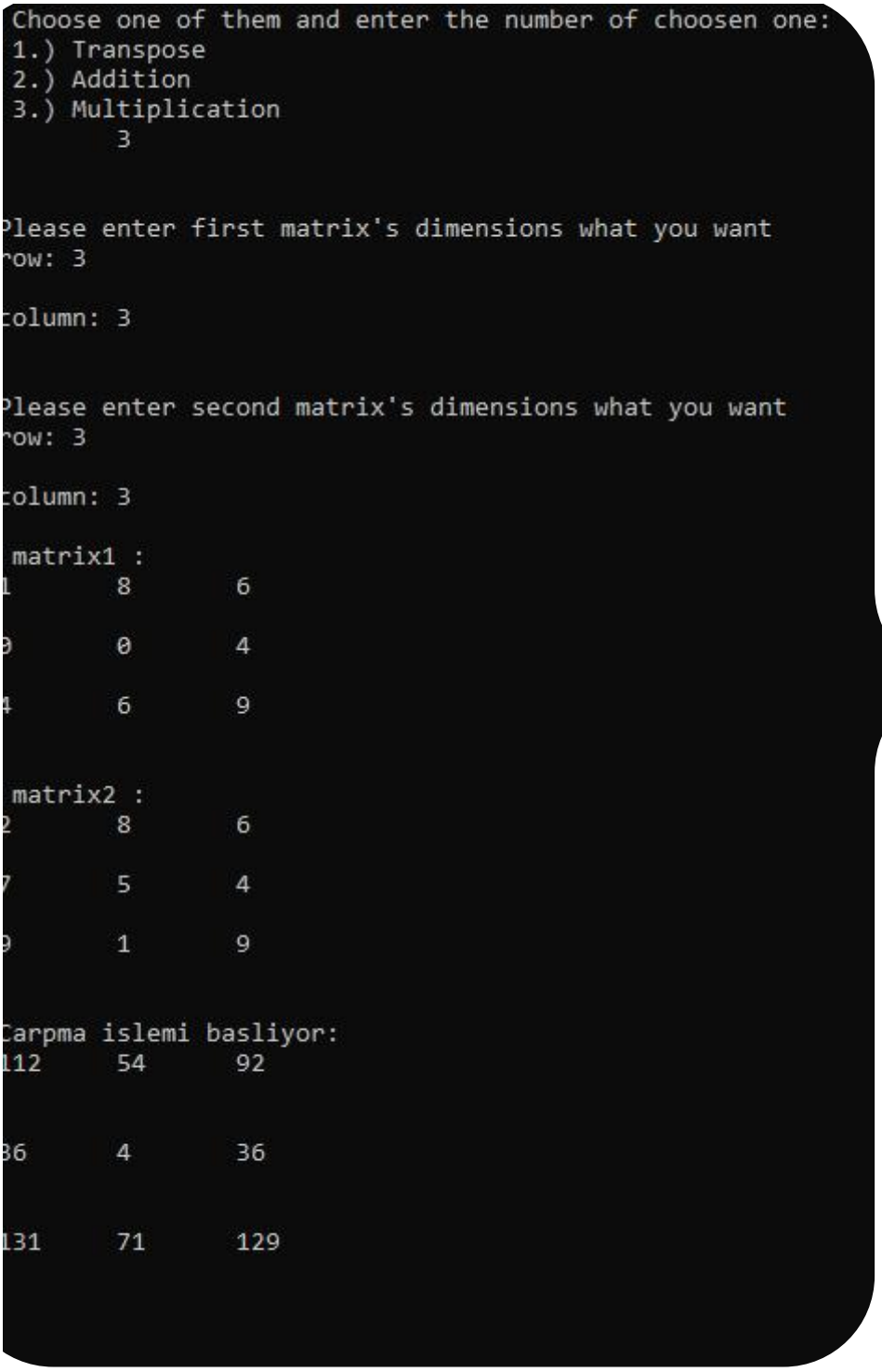
And save the data from user about rows and columns.

3.Four FOR Loops Are using for generate random matrices values with dimensions from user.

4.Last two FOR Loops Are using for printing generated matrices on screen.

5.Next two FOR Loops Are using for calculate the Multiplication.

4.1-MULTIPLICATION OUTPUT



1.Choosing Multiplication Function with write ‘3’.

2.Entering matrices’s dimensions.

3.Program prints “matrix1”and “matrix2”.

5. Program prints “matrix1” * “matrix2”.