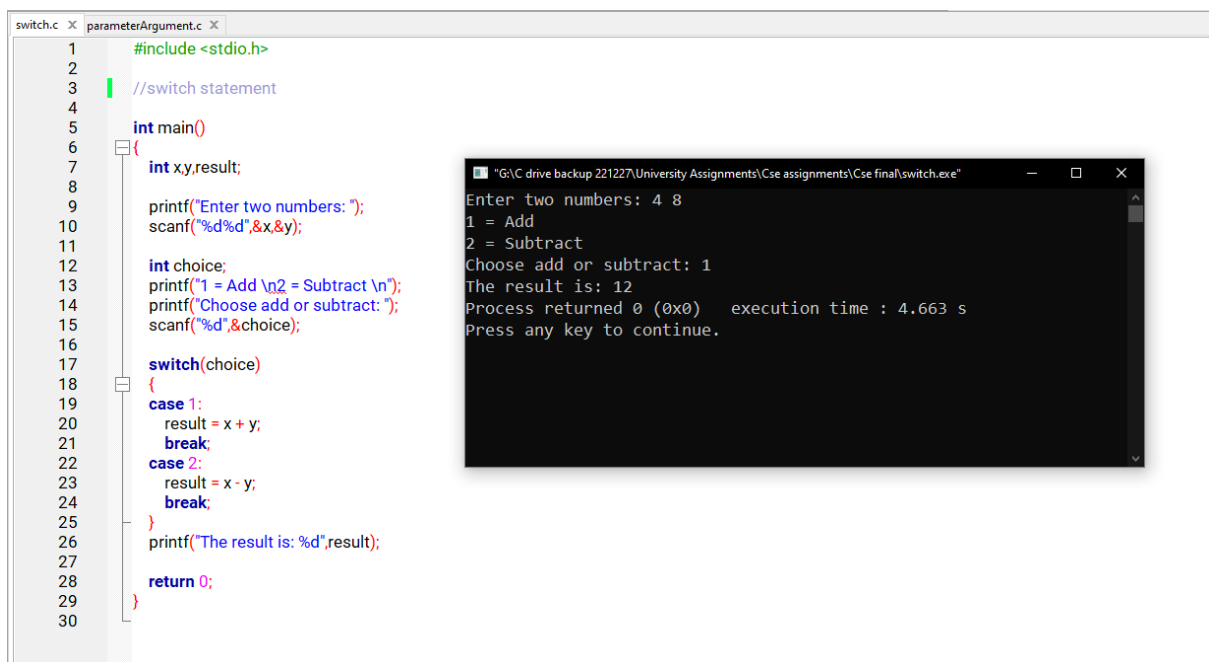


CSE notes for final

If you find any correction, please contact: Akib Reza 1061

Switch:



The image shows a C program in a code editor and its execution output in a terminal window.

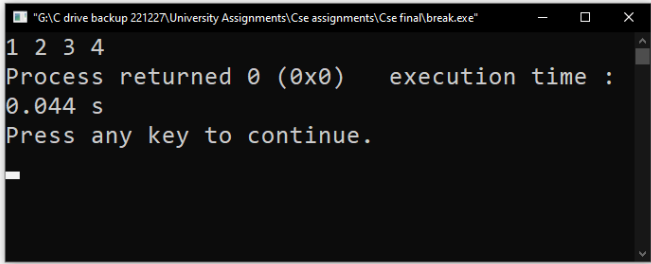
Code Editor (switch.c):

```
1  #include <stdio.h>
2
3  //switch statement
4
5  int main()
6  {
7      int x,y,result;
8
9      printf("Enter two numbers: ");
10     scanf("%d%d",&x,&y);
11
12     int choice;
13     printf("1 = Add \n2 = Subtract \n");
14     printf("Choose add or subtract: ");
15     scanf("%d",&choice);
16
17     switch(choice)
18     {
19     case 1:
20         result = x + y;
21         break;
22     case 2:
23         result = x - y;
24         break;
25     }
26     printf("The result is: %d",result);
27
28     return 0;
29 }
30
```

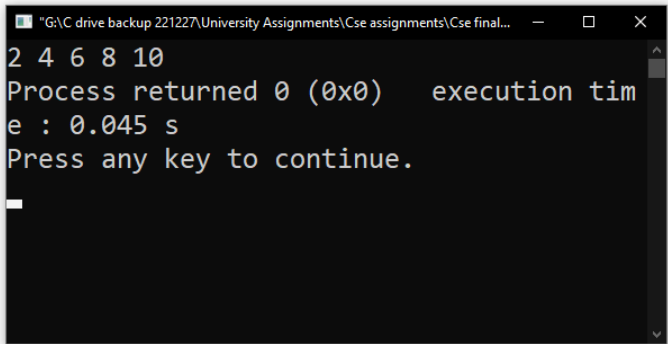
Terminal Output:

```
"G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\switch.exe"
Enter two numbers: 4 8
1 = Add
2 = Subtract
Choose add or subtract: 1
The result is: 12
Process returned 0 (0x0)   execution time : 4.663 s
Press any key to continue.
```

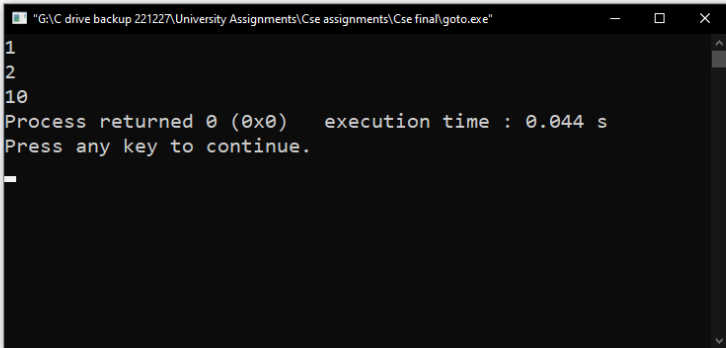
```
1 #include <stdio.h>
2
3 //break statement
4
5 int main() {
6     int i;
7     for (i = 1; i <= 10; i++) {
8         if (i == 5) {
9             break;
10        }
11        printf("%d ", i);
12    }
13    return 0;
14 }
15
```



```
1 #include <stdio.h>
2
3 //continue statement
4
5 int main() {
6     int i;
7     for (i = 1; i <= 10; i++) {
8         if (i % 2 != 0) {
9             continue;
10        }
11        printf("%d ", i);
12    }
13    return 0;
14 }
15
```



```
1 #include <stdio.h>
2
3 //goto statement
4
5 int main()
6 {
7     printf("1\n");
8     printf("2\n");
9
10    goto jumpLine;
11
12    printf("3\n");
13    printf("4\n");
14
15    jumpLine:
16    printf("10");
17
18    return 0;
19 }
20
```



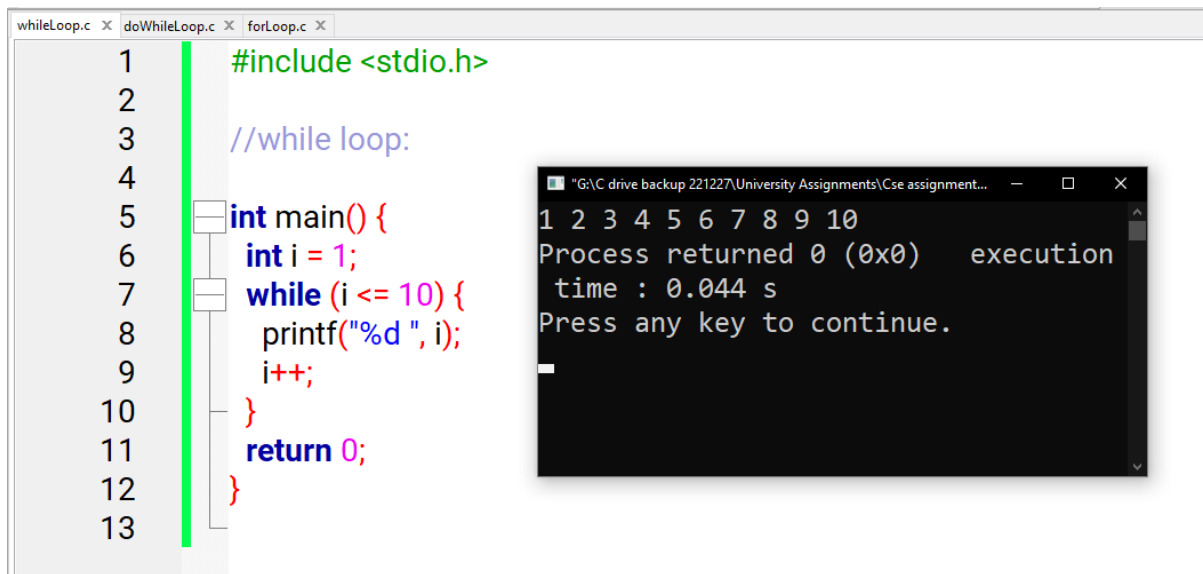
The main difference between while, do while and for loop is:

While loop: The loop body is executed as long as the condition is true. If the condition is false initially, the loop body is never executed.

Do while loop: The loop body is executed at least once, and then the condition is checked. If the condition is true, the loop body is executed again.

For loop: It consists of a loop variable initialization, a termination condition, and an increment/decrement. The loop body is executed as long as the termination condition is true.

Example:

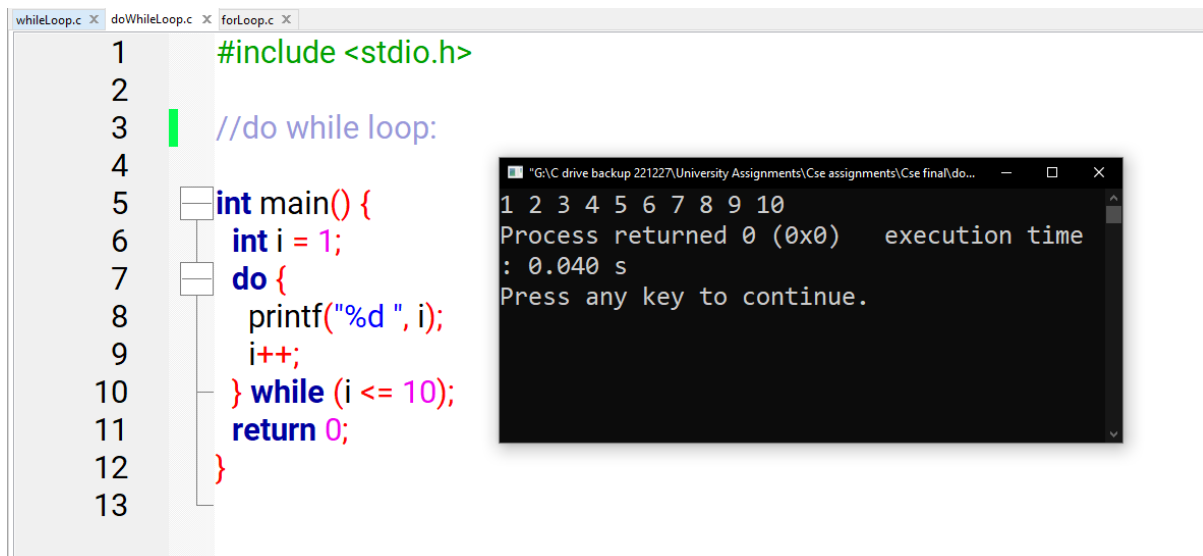


The screenshot shows a C program in a text editor with three tabs: `whileLoop.c`, `doWhileLoop.c`, and `forLoop.c`. The `whileLoop.c` tab is active, displaying the following code:

```
1 #include <stdio.h>
2
3 //while loop:
4
5 int main() {
6     int i = 1;
7     while (i <= 10) {
8         printf("%d ", i);
9         i++;
10    }
11    return 0;
12 }
13
```

To the right of the code editor is a terminal window titled `"G:\C drive backup 221227\University Assignments\Cse assignment..."`. It displays the output of the program:

```
1 2 3 4 5 6 7 8 9 10
Process returned 0 (0x0)   execution
time : 0.044 s
Press any key to continue.
```

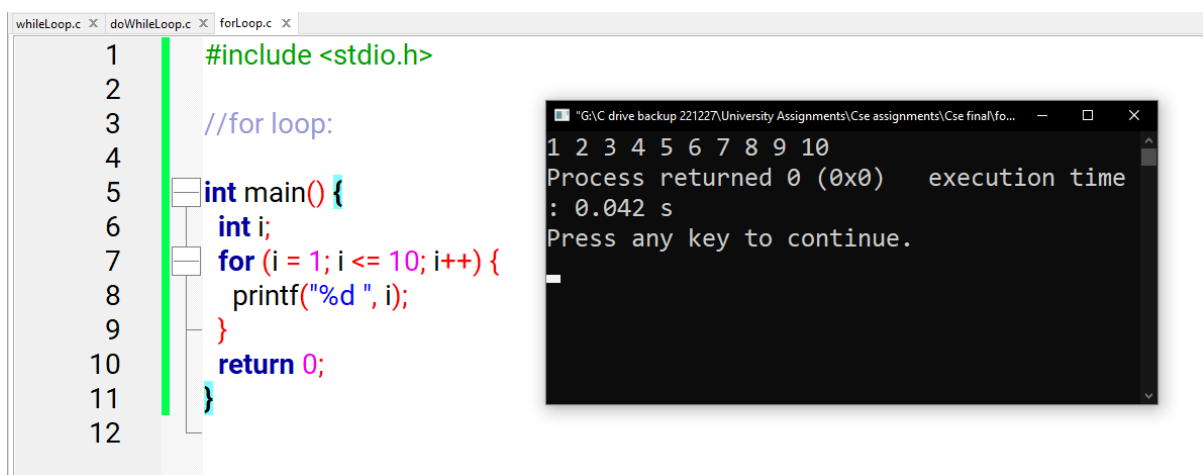


The screenshot shows the same C program in a text editor, but now the `doWhileLoop.c` tab is active. The code is as follows:

```
1 #include <stdio.h>
2
3 //do while loop:
4
5 int main() {
6     int i = 1;
7     do {
8         printf("%d ", i);
9         i++;
10    } while (i <= 10);
11    return 0;
12 }
13
```

The terminal window to the right, titled `"G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\do..."`, shows the output:

```
1 2 3 4 5 6 7 8 9 10
Process returned 0 (0x0)   execution time
: 0.040 s
Press any key to continue.
```



The screenshot shows the same C program in a text editor, with the `forLoop.c` tab now active. The code is as follows:

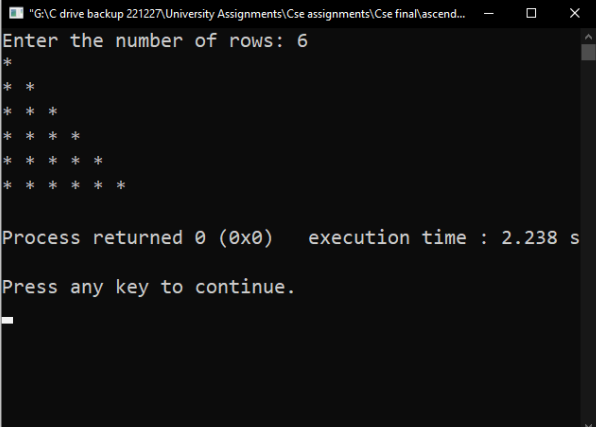
```
1 #include <stdio.h>
2
3 //for loop:
4
5 int main() {
6     int i;
7     for (i = 1; i <= 10; i++) {
8         printf("%d ", i);
9     }
10    return 0;
11 }
12
```

The terminal window to the right, titled `"G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\fo..."`, shows the output:

```
1 2 3 4 5 6 7 8 9 10
Process returned 0 (0x0)   execution time
: 0.042 s
Press any key to continue.
```

Triangle Pattern:

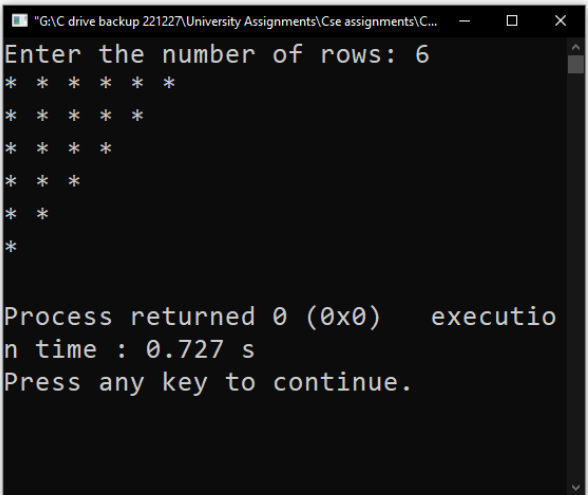
```
ascendingTriangle.c x descendingTriangle.c x
1      #include <stdio.h>
2
3      /*
4      /* *
5      /* * *
6      /* * * *
7      /* * * * *
8
9      int main() {
10         int i, j, rows;
11         printf("Enter the number of rows: ");
12         scanf("%d", &rows);
13         for (i = 1; i <= rows; i++) {
14             for (j = 1; j <= i; j++) {
15                 printf("* ");
16             }
17             printf("\n");
18         }
19         return 0;
20     }
21
22
```



```
"G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\ascend...
Enter the number of rows: 6
*
* *
* * *
* * * *
* * * * *
* * * * *

Process returned 0 (0x0)   execution time : 2.238 s
Press any key to continue.
```

```
ascendingTriangle.c x descendingTriangle.c x
1      #include <stdio.h>
2
3      /* * * * * *
4      /* * * *
5      /* * *
6      /* *
7      /*
8
9      int main() {
10         int i, j, rows;
11         printf("Enter the number of rows: ");
12         scanf("%d", &rows);
13         for (i = 1; i <= rows; i++) {
14             for (j = rows; j >= i; j--) {
15                 printf("* ");
16             }
17             printf("\n");
18         }
19         return 0;
20     }
21
22
```



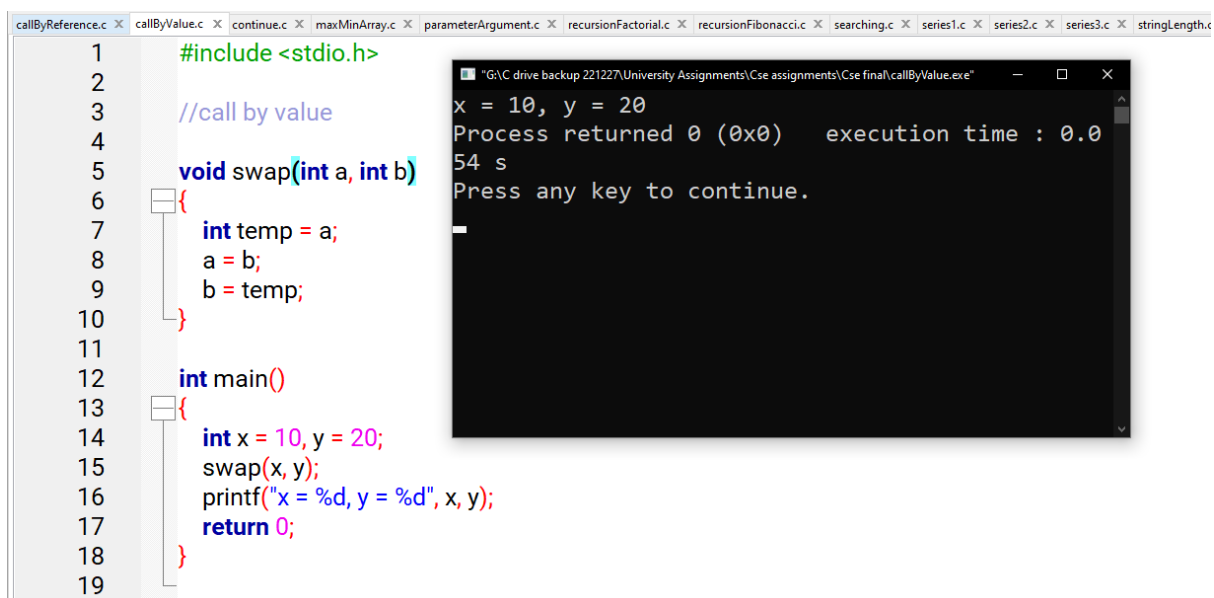
```
"G:\C drive backup 221227\University Assignments\Cse assignments\C...
Enter the number of rows: 6
* * * * *
* * * *
* * *
* *
*
*

Process returned 0 (0x0)   executio
n time : 0.727 s
Press any key to continue.
```

Call by value and Call by reference:

Call by value: In this method, a copy of the actual argument is passed to the formal parameter. Changes made to the formal parameter don't affect the actual argument.

Call by reference: In this method, a reference (memory address) to the actual argument is passed to the formal parameter. Changes made to the formal parameter directly affect the actual argument.

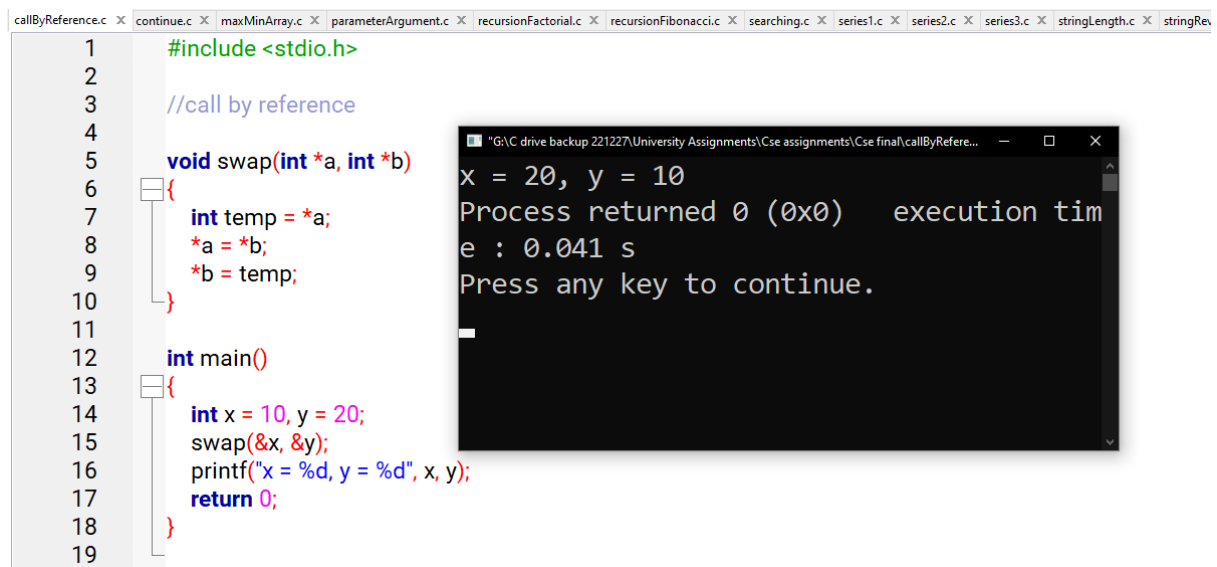


The screenshot shows a C++ IDE with a tab labeled 'callByValue.c'. The code is as follows:

```
1 #include <stdio.h>
2
3 //call by value
4
5 void swap(int a, int b)
6 {
7     int temp = a;
8     a = b;
9     b = temp;
10 }
11
12 int main()
13 {
14     int x = 10, y = 20;
15     swap(x, y);
16     printf("x = %d, y = %d", x, y);
17     return 0;
18 }
19
```

Overlaid on the code is a terminal window titled "G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\callByValue.exe". The terminal output is:

```
x = 10, y = 20
Process returned 0 (0x0)   execution time : 0.054 s
Press any key to continue.
```



The screenshot shows a C++ IDE with a tab labeled 'callByReference.c'. The code is as follows:

```
1 #include <stdio.h>
2
3 //call by reference
4
5 void swap(int *a, int *b)
6 {
7     int temp = *a;
8     *a = *b;
9     *b = temp;
10 }
11
12 int main()
13 {
14     int x = 10, y = 20;
15     swap(&x, &y);
16     printf("x = %d, y = %d", x, y);
17     return 0;
18 }
19
```

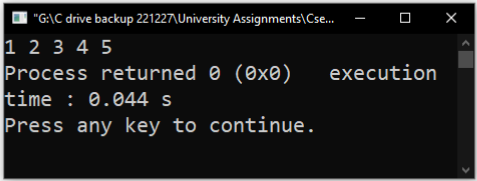
Overlaid on the code is a terminal window titled "G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\callByRefere...". The terminal output is:

```
x = 20, y = 10
Process returned 0 (0x0)   execution time : 0.041 s
Press any key to continue.
```

Series print with for loop examples:

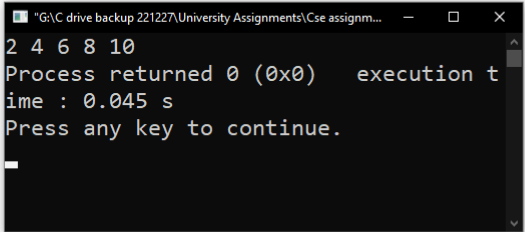
```
maxMinArray.c x parameterArgument.c x recursionFactorial.c x recursionFibonacci.c x searching.c x series1.c x series2.c x series3.c x stringLength.c x stringReverse.c x structure.c x sum.c x
```

```
1  #include <stdio.h>
2
3  // Use loop to write this series: 1, 2,3,4, 5
4
5  int main() {
6      int i;
7      for (i = 1; i <= 5; i++) {
8          printf("%d ", i);
9      }
10     return 0;
11 }
12
```



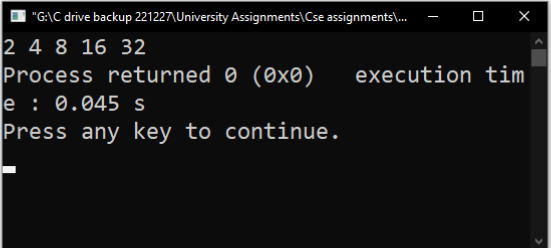
```
maxMinArray.c x parameterArgument.c x recursionFactorial.c x recursionFibonacci.c x searching.c x series2.c x series3.c x stringLength.c x stringReverse.c x structure.c x sum.c x switch.c x to
```

```
1  #include <stdio.h>
2
3  // Use loop to write this series: 2,4,6,8,10
4
5  int main() {
6      int i;
7      for (i = 2; i <= 10; i = i+2) {
8          printf("%d ", i);
9      }
10     return 0;
11 }
12
13
```



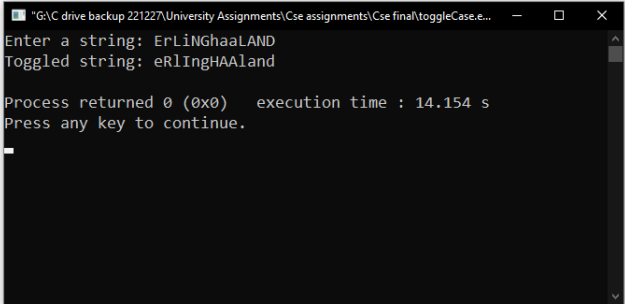
```
maxMinArray.c x parameterArgument.c x recursionFactorial.c x recursionFibonacci.c x searching.c x series3.c x stringLength.c x stringReverse.c x structure.c x sum.c x switch.c x toggleCase.c
```

```
1  #include <stdio.h>
2
3  // Use loop to write this series: 2,4,8,16,32
4
5  int main() {
6      int i;
7      for (i = 2; i <= 32; i = i*2) {
8          printf("%d ", i);
9      }
10     return 0;
11 }
12
```



Array:

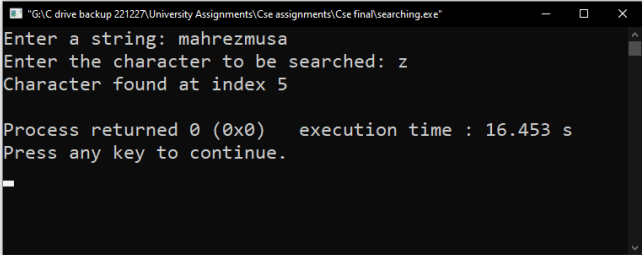
```
maxMinArray.c x parameterArgument.c x recursionFactorial.c x recursionFibonacci.c x searching.c x stringLength.c x stringReverse.c x structure.c x sum.c x switch.c x toggleCase.c x
1      #include <stdio.h>
2      #include <string.h>
3
4      //toggle case in array
5
6      int main() {
7          char str[100];
8          int i;
9          printf("Enter a string: ");
10         scanf("%s", str);
11         for (i = 0; i < strlen(str); i++) {
12             if (str[i] >= 'a' && str[i] <= 'z') {
13                 str[i] = str[i] - 32;
14             } else if (str[i] >= 'A' && str[i] <= 'Z') {
15                 str[i] = str[i] + 32;
16             }
17         }
18         printf("Toggled string: %s\n", str);
19         return 0;
20     }
21
```



```
"G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\toggleCase.e...
Enter a string: ErLiNGhaaLAND
Toggled string: eRLiNGHAAland

Process returned 0 (0x0)   execution time : 14.154 s
Press any key to continue.
```

```
maxMinArray.c x parameterArgument.c x recursionFactorial.c x recursionFibonacci.c x searching.c x stringLength.c x stringReverse.c x structure.c x sum.c x switch.c x
1      #include <stdio.h>
2      #include <string.h>
3
4      //character search in array
5
6      int main() {
7          char str[100], x;
8          int i, flag = 0;
9          printf("Enter a string: ");
10         scanf("%s", str);
11         printf("Enter the character to be searched: ");
12         scanf("%c", &x);
13         for (i = 0; i < strlen(str); i++) {
14             if (str[i] == x) {
15                 flag = 1;
16                 break;
17             }
18         }
19         if (flag == 1) {
20             printf("Character found at index %d\n", i);
21         } else {
22             printf("Character not found\n");
23         }
24         return 0;
25     }
26
```



```
"G:\C drive backup 221227\University Assignments\Cse assignments\Cse final\searching.exe
Enter a string: mahrezmusa
Enter the character to be searched: z
Character found at index 5

Process returned 0 (0x0)   execution time : 16.453 s
Press any key to continue.
```

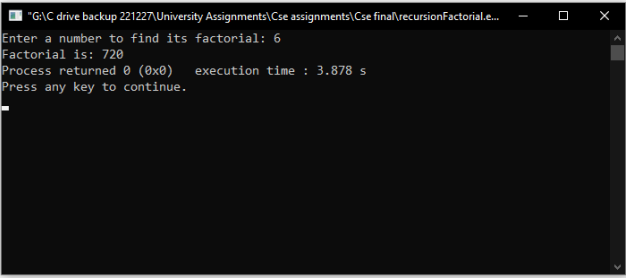


```
recursionFactorial.c × recursionFibonacci.c × stringLength.c × stringReverse.c × structure.c × switch.c × parameterArgument.c × maxMinArray.c × *sum.c ×
1 #include <stdio.h>
2
3 //summation and average of numbers in array
4
5 int main() {
6     int n, i, sum = 0;
7     printf("Enter the number of elements in the array: ");
8     scanf("%d", &n);
9
10    int a[n];
11    printf("Enter the elements of the array: ");
12
13    for (i = 0; i < n; i++) {
14        scanf("%d", &a[i]);
15    }
16
17    for (i = 0; i < n; i++) {
18        sum = sum + a[i];
19    }
20
21    printf("Sum of the elements in the array: %d\n", sum);
22
23    float average = (float)sum / n;
24    printf("Average: %.2f", average);
25    return 0;
26 }
27
```

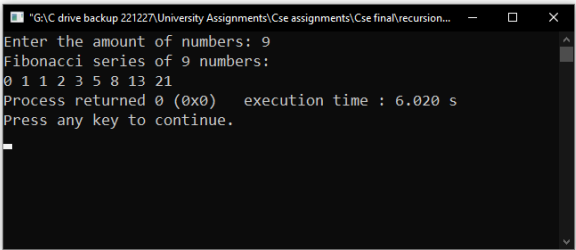
```
recursionFactorial.c × recursionFibonacci.c × stringLength.c × stringReverse.c × structure.c × switch.c × parameterArgument.c × maxMinArray.c ×
1 #include <stdio.h>
2
3 //maximum and minimum of array
4
5 int main() {
6     int n, i;
7     int max, min;
8
9     printf("Enter the number of elements in the array: ");
10    scanf("%d", &n);
11
12    int a[n];
13    printf("Enter the elements of the array: ");
14
15    for (i = 0; i < n; i++) {
16        scanf("%d", &a[i]);
17    }
18    max = a[0];
19    min = a[0];
20    for (i = 1; i < n; i++) {
21        if (a[i] > max) {
22            max = a[i];
23        }
24        if (a[i] < min) {
25            min = a[i];
26        }
27    }
28    printf("Maximum element in the array: %d\n", max);
29    printf("Minimum element in the array: %d\n", min);
30    return 0;
31 }
32
33
```

Recursion:

```
recursionFactorial.c x recursionFibonacci.c x stringLength.c x stringReverse.c x structure.c x switch.c x parameterArgument.c x
1 #include <stdio.h>
2
3 //factorial using recursion
4
5 int factorial(int n)
6 {
7     if (n == 0)
8         return 1;
9     return n * factorial(n - 1);
10 }
11
12 int main()
13 {
14     int num;
15     printf("Enter a number to find its factorial: ");
16     scanf("%d", &num);
17     printf("Factorial is: %d", factorial(num));
18     return 0;
19 }
20
```

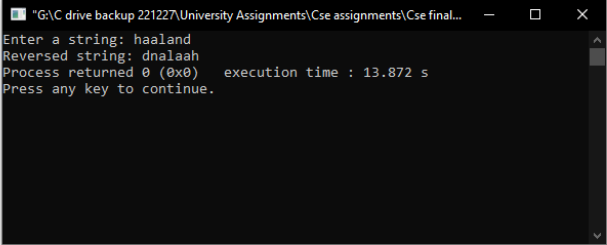


```
recursionFibonacci.c x stringLength.c x stringReverse.c x structure.c x switch.c x parameterArgument.c x
1 #include <stdio.h>
2
3 //fibonacci series using recursion
4
5 int fibonacci(int n)
6 {
7     if (n == 0)
8         return 0;
9     if (n == 1)
10        return 1;
11    return fibonacci(n - 1) + fibonacci(n - 2);
12 }
13
14 int main()
15 {
16     int num, i;
17     printf("Enter the amount of numbers: ");
18     scanf("%d", &num);
19     printf("Fibonacci series of %d numbers: \n", num);
20     for (i = 0; i < num; i++)
21     {
22         printf("%d ", fibonacci(i));
23     }
24     return 0;
25 }
26
```

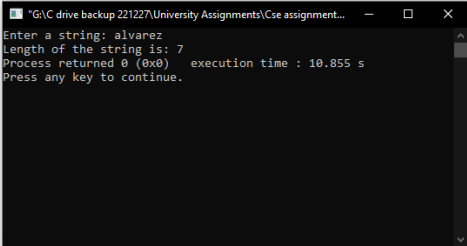


String:

```
stringLength.c x stringReverse.c x structure.c x switch.c x parameterArgument.c x
1      #include <stdio.h>
2      #include <string.h>
3
4      //string reverse with strev
5
6      int main() {
7          char str[100];
8          printf("Enter a string: ");
9          gets(str);
10         strev(str);
11         printf("Reversed string: %s", str);
12         return 0;
13     }
14
```



```
stringLength.c x structure.c x switch.c x parameterArgument.c x
1      #include <stdio.h>
2      #include <string.h>
3
4      //string length with strlen
5
6      int main() {
7          char str[100];
8          int length;
9          printf("Enter a string: ");
10         gets(str);
11         length = strlen(str);
12         printf("Length of the string is: %d", length);
13         return 0;
14     }
15
```



Structure:

```
structure.c x switch.c x parameterArgument.c x
1      #include <stdio.h>
2      #include <string.h>
3
4      //structure example
5
6      struct student {
7          int roll;
8          int age;
9          float cgpa;
10     };
11
12     int main() {
13         struct student s1;
14         s1.roll = 1061;
15         s1.age = 20;
16         s1.cgpa = 2.55;
17
18         printf("Student Details: \n");
19         printf("Roll: %d\n", s1.roll);
20         printf("Age: %d\n", s1.age);
21         printf("CGPA: %.2f\n", s1.cgpa);
22
23         return 0;
24     }
25
```

