

## C Programming Day 10

### Task 9

Name:Sargam Gupta  
Roll No:FYIT1-19

#### Code 1

The screenshot shows a code editor interface with a dark theme. On the left, there is a sidebar with various language icons: Python, Go, C, C++, JavaScript, TypeScript, and others. The main area has tabs for "main.c" and "Output". The code in "main.c" is as follows:

```
1 #include <stdio.h>
2
3 int main() {
4     int a, b, c;
5
6     printf("Enter three numbers: ");
7     scanf("%d %d %d", &a, &b, &c);
8
9
10    if (a >= b && a >= c)
11        printf("Largest number is: %d\n", a);
12    else if (b >= a && b >= c)
13        printf("Largest number is: %d\n", b);
14    else
15        printf("Largest number is: %d\n", c);
16
17    return 0;
18 }
19
20
21
```

The "Output" tab shows the result of running the code with inputs 3, 4, and 5. The output is:

```
Enter three numbers: 3 4 5
Largest number is: 5
--- Code Execution Successful ---
```

#### Code 2

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing a C program named `main.c`. The code swaps two integers, `a` and `b`, using a temporary variable `temp`. On the right is the output pane, which displays the execution results.

```
main.c
1 // C Program to swap two numbers using a temporary variable
2
3 #include<stdio.h>
4 int main(){
5     int a,b,temp;
6
7     printf("Enter two numbers:");
8     scanf("%d %d",&a,&b);
9
10    temp=a;
11    a=b;
12    b=temp;
13
14
15    printf("After swapping:\n");
16    printf("a=%d\n",a);
17    printf("b=%d\n",b);
18    return 0;
19 }
20
```

Output

```
Enter two numbers:4 5
After swapping:
a=5
b=4

*** Code Execution Successful ***
```

## Code 3

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing a C program named `main.c`. The program prompts the user for a lower limit, an upper limit, and a number, then checks if the number is within the specified range. On the right is the output pane, which displays the execution results.

```
main.c
1 //find whether the number is in range or not
2 #include <stdio.h>
3
4 int main() {
5     int lower, upper, num;
6
7     printf("Enter lower limit and upper limit: "); scanf("%d %d", &lower, &upper);
8
9     printf("Enter a number: "); scanf("%d", &num);
10
11    if (num >= lower && num <= upper) printf("Number is within the range");
12    else
13        printf("Number is outside the range");
14
15    return 0;
16 }
```

Output

```
Enter lower limit and upper limit: 10 20
Enter a number: 15
Number is within the range

*** Code Execution Successful ***
```

## Code 4

main.c

```
1 #include <stdio.h>
2
3 int main() {
4     int i;
5     float f;
6
7     printf("Enter an integer: ");
8     scanf("%d", &i);
9
10    float floatValue = i;
11
12
13
14    printf("Enter a float number: ");
15    scanf("%f", &f);
16
17
18    int intValue = f;
19
20
21    printf("\ninteger to Float conversion:");
22    printf("\n\tInteger = %d, Converted Float = %.2f", i, floatValue);
23
24
25    printf("\n\nfloat to Integer conversion:");
26    printf("\n\tfloat = %.2f, Converted Integer = %d", f, intValue);
27
28    return 0;
29 }
30
31
32
```

Output

```
Enter an integer: 4
Enter a float number: 5.00

Integer to Float conversion:
Integer = 4, Converted Float = 4.00

Float to Integer conversion:
Float = 5.00, Converted Integer = 5

*** Code Execution Successful ***
```

## Code 5

main.c

```
1 //find whether the number is positive or negative
2 #include <stdio.h>
3
4 int main() { int num;
5     printf("Enter a number: "); scanf("%d", &num);
6
7     if(num > 0)
8         printf("Positive number");
9     else if(num < 0)
10        printf("Negative number");
11     else
12        printf("Zero");
13 }
```

Output

```
Enter a number: 6
Positive number

*** Code Execution Successful ***
```

## Code 6

```
main.c | Run | Output | Clear
```

```
1 //find size of functions
2 #include <stdio.h>
3
4 int main() {
5     printf("Size of int: %lu bytes\n", sizeof(int)); printf("Size of float: %lu bytes\n", sizeof
6         (float)); printf("Size of double: %lu bytes\n", sizeof(double)); printf("Size of char: %lu
7         bytes\n", sizeof(char)); return 0;
}
```

Output

```
Size of int: 4 bytes
Size of float: 4 bytes
Size of double: 8 bytes
Size of char: 1 byte
==== Code Execution Successful ===
```

## Code 7

```
main.c | Run | Output | Clear
```

```
1 //basic calculator
2 #include <stdio.h>
3
4 int main() {
5     int choice; float a, b;
6
7     while(1) { printf("\n1.Add\n2.Subtract\n3.Multiply\n4.Divide\n5.Exit\n"); printf("Enter
8         choice: ");
9     scanf("%d", &choice);
10
11    if(choice == 5) break;
12
13    switch(choice) {
14        case 1: printf("Result = %.2f", a + b); break; case 2: printf("Result = %.2f", a - b); break;
15        case 3: printf("Result = %.2f", a * b); break; case 4: printf("Result = %.2f", a / b);
16        break;
17    }
18
19    return 0;
20
21
22
```

Output

```
1.Add
2.Subtract
3.Multiply
4.Divide
5.Exit
Enter choice: 1
Enter two numbers: 3 2
Result = 5.00
1.Add
2.Subtract
3.Multiply
4.Divide
5.Exit
Enter choice: 2
Enter two numbers: 5 3
Result = 2.00
1.Add
2.Subtract
3.Multiply
4.Divide
5.Exit
Enter choice: 3
Enter two numbers: 5 3
Result = 15.00
1.Add
2.Subtract
3.Multiply
4.Divide
5.Exit
Enter choice: 5
```

## Code 8

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing `main.c`. The code defines a function `gcd` that returns the greatest common divisor of two integers `a` and `b`. It also calculates the least common multiple (`lcm`) of the two numbers. The main function prompts the user to enter two numbers, reads them, and then prints the GCD and LCM. The right side of the interface is the output pane, which displays the execution results. The user entered 5 and 4, and the program output is:

```
Enter two numbers: 5 4
GCD = 1
LCM = 20
==== Code Execution Successful ===
```

## Code 9

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing `main.c`. The code demonstrates string manipulation and comparison. It first asks for two strings, concatenates them, and then copies the second string to the first. It then compares the two strings using `strcmp`. The right side of the interface is the output pane, which displays the execution results. The user entered "sar" and "gam", and the program output is:

```
Enter first string: sar
Enter second string: gam
Length of s1 = 3
Copy s2 to s1 = gam
Concatenation = gamgam
Comparison = 103
==== Code Execution Successful ===
```

## Code 10

The screenshot shows a dark-themed online compiler interface. On the left, there's a sidebar with icons for various languages: Python, C, C++, Java, JavaScript, TypeScript, Go, Node.js, and Docker. The main area has tabs for "main.c" and "Output". The code in "main.c" is:

```
1 #include <stdio.h>
2
3 struct Book { char title[30];
4 char author[30]; float price;
5 };
6
7 int main() {
8 struct Book b[2] = {
9 {"C Programming", "Dennis", 450},
10 {"Data Structures", "Mark", 550}
11 };
12
13 for(int i = 0; i < 2; i++) {
14 printf("\nTitle: %s\nAuthor: %s\nPrice: %.2f\n", b[i].title, b[i].author, b[i].price);
15 }
16 return 0;
17 }
```

The "Output" tab shows the results of running the code:

```
Title: C Programming
Author: Dennis
Price: 450.00

Title: Data Structures
Author: Mark
Price: 550.00

--- Code Execution Successful ---
```

## Code 11

The screenshot shows a dark-themed online compiler interface. On the left, there's a sidebar with icons for various languages: Python, C, C++, Java, JavaScript, TypeScript, Go, Node.js, and Docker. The main area has tabs for "main.c" and "Output". The code in "main.c" is:

```
1 //swap using external type
2 #include <stdio.h>
3
4 int main() {
5 int a[5] = {1,2,3,4,5};
6 int start = 0, end = 4, temp;
7
8 while(start < end) { temp = a[start]; a[start] = a[end]; a[end] = temp; start++; end--;}
9
10 for(int i = 0; i < 5; i++) printf("%d ", a[i]);
11
12 return 0;
13 }
```

The "Output" tab shows the results of running the code:

```
5 4 3 2 1

--- Code Execution Successful ---
```

## Code 12

The screenshot shows a code editor interface with a dark theme. On the left, the code file 'main.c' is displayed:

```
main.c
1 //find sum using array
2 #include <stdio.h>
3
4 int sumArray(int a[], int n) { int sum = 0;
5 for(int i = 0; i < n; i++) sum += a[i];
6 return sum;
7 }
8
9 int main() {
10 int a[5] = {1,2,3,4,5};
11 printf("Sum = %d", sumArray(a,5)); return 0;
12 }
13
14 }
```

At the top right, there are several icons: a copy button, a brightness slider, a share icon, and a 'Run' button. To the right of the run button is an 'Output' section. The output shows the result of running the code:

Sum = 15  
\*\*\* Code Execution Successful \*\*\*

### Code 13

The screenshot shows a code editor interface with a dark theme. On the left, the code file 'main.c' is displayed:

```
main.c
1 //Left rotate array by k positions
2 #include <stdio.h>
3
4 int main() {
5 int a[5] = {1,2,3,4,5}, k = 2, temp;
6
7 for(int i = 0; i < k; i++) { temp = a[0];
8 for(int j = 0; j < 4; j++) a[j] = a[j+1];
9 a[4] = temp;
10 }
11 for(int i = 0; i < 5; i++) printf("%d ", a[i]);
12
13 return 0;
14 }
```

At the top right, there are several icons: a copy button, a brightness slider, a share icon, and a 'Run' button. To the right of the run button is an 'Output' section. The output shows the result of running the code:

3 4 5 1 2  
\*\*\* Code Execution Successful \*\*\*

### Code 14

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing `main.c`. The code is a C program that checks if two 2D arrays are equal. It includes #include <stdio.h>, defines arrays `a[2][2]` and `b[2][2]`, initializes them with values {1,2},{3,4} and {1,2},{3,4} respectively, and then iterates through them to check if they are equal. If they are not equal, it sets a flag to 1 and prints "Arrays are not equal". Otherwise, it prints "Arrays are equal". The code editor has standard icons for copy, paste, share, and run. The run button is highlighted in blue. To the right is the output pane which displays the results of the execution. It shows the output "Arrays are equal" followed by the message "Code Execution Successful".

```
main.c
1 //Check if two 2D arrays are equal
2 #include <stdio.h>
3 int main() {
4     int a[2][2] = {{1,2},{3,4}};
5     int b[2][2] = {{1,2},{3,4}};
6     int flag = 1;
7
8     for(int i = 0; i < 2; i++) for(int j = 0; j < 2; j++)
9         if(a[i][j] != b[i][j]) flag = 0;
10
11    if(flag)
12        printf("Arrays are equal");
13    else
14        printf("Arrays are not equal");
15
16    return 0;
17}
```

## Code 15

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing `main.c`. The code is a C program that counts the number of blank spaces in a string entered by the user. It includes #include <stdio.h>, declares a character array `str[100]` and an integer `count` initialized to 0. It prompts the user to enter a string using `printf` and reads the input using `fgets` from `stdin`. Then, it loops through each character of the string. If a character is a blank space (' '), it increments the `count`. Finally, it prints the total number of blank spaces using `printf` and returns 0. The code editor has standard icons for copy, paste, share, and run. The run button is highlighted in blue. To the right is the output pane which displays the results of the execution. It shows the prompt "Enter a string: Hello World this is C program" followed by the output "Total blank spaces: 5" and the message "Code Execution Successful".

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     char str[100];
5     int count = 0;
6
7     printf("Enter a string: ");
8     // fgets allows spaces in the input, unlike scanf
9     fgets(str, sizeof(str), stdin);
10
11    for (int i = 0; str[i] != '\0'; i++) {
12        if (str[i] == ' ') {
13            count++;
14        }
15    }
16
17    printf("Total blank spaces: %d\n", count);
18    return 0;
19 }
20
```

## Code 16

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing the following C code:

```
main.c
1 #include <stdio.h>
2
3 int main(void)
4 {
5     int a[2][2] = { {1, 2}, {1, 2} };
6     int visited[2][2] = { {0, 0}, {0, 0} };
7
8     for (int i = 0; i < 2; i++) {
9         for (int j = 0; j < 2; j++) {
10            if (visited[i][j])
11                continue;
12
13            int count = 1;
14
15            for (int x = i; x < 2; x++) {
16                int ystart = (x == i) ? (j + 1) : 0;
17                for (int y = ystart; y < 2; y++) {
18                    if (a[i][j] == a[x][y]) {
19                        count++;
20                        visited[x][y] = 1;
21                    }
22                }
23            }
24
25            printf("%d occurs %d times\n", a[i][j], count);
26        }
27    }
28
29    return 0;
30 }
```

On the right is the output pane, which displays the results of the execution:

```
Output
1 occurs 2 times
2 occurs 2 times
--- Code Execution Successful ---
```

## Code 17

The screenshot shows a code editor interface with a dark theme. On the left is the code editor pane containing the following C code:

```
main.c
1 //using bubble sort, sort in descending order
2 #include <stdio.h>
3
4 void bubbleSort(int a[], int n) { for(int i = 0; i < n-1; i++)
5     for(int j = 0; j < n-i-1; j++) if(a[j] < a[j+1]) {
6         int t = a[j]; a[j] = a[j+1]; a[j+1] = t;
7     }
8
9
10 int main() {
11     int a[5] = {3,1,5,2,4};
12     bubbleSort(a,5);
13
14     for(int i = 0; i < 5; i++) printf("%d ", a[i]);
15
16     return 0;
17 }
```

On the right is the output pane, which displays the results of the execution:

```
Output
5 4 3 2 1
--- Code Execution Successful ---
```

## Code 18

The screenshot shows a code editor interface with a dark theme. On the left, the code file 'main.c' contains the following C code:

```
1 //Count vowels and consonants
2 #include <stdio.h>
3 int main() {
4     int decimal, binary[32], i = 0;
5
6     printf("Enter a decimal number: ");
7     scanf("%d", &decimal);
8
9     if (decimal == 0) { printf("Binary number: 0"); 
10    return 0;
11 }
12 while (decimal > 0) { binary[i] = decimal % 2; decimal = decimal / 2; i++;
13 }
14
15 printf("Binary number: "); for (int j = i - 1; j >= 0; j--) {
16    printf("%d", binary[j]);
17 }
18
19 return 0;
20 }
21 }
```

At the top right, there are several icons: a refresh symbol, a brightness control, a share icon, and a 'Run' button. To the right of the code area is the 'Output' panel, which displays the results of the execution:

```
Enter a decimal number: 3.67
Binary number: 11
*** Code Execution Successful ***
```

## Code 19

The screenshot shows a code editor interface with a dark theme. On the left, the code file 'main.c' contains the following C code:

```
1 //give max and min no. from the array
2 #include <stdio.h>
3 int main() {
4     int a[5] = {4,2,9,12,16};
5     int max = a[0], min = a[0];
6
7     for(int i = 1; i < 5; i++) { if(a[i] > max) max = a[i];
8         if(a[i] < min) min = a[i];
9     }
10
11    printf("Max = %d\nMin = %d", max, min); return 0;
12 }
13 }
```

At the top right, there are several icons: a refresh symbol, a brightness control, a share icon, and a 'Run' button. To the right of the code area is the 'Output' panel, which displays the results of the execution:

```
Max = 16
Min = 2
*** Code Execution Successful ***
```

## Code 20

The screenshot shows a code editor interface with a dark theme. On the left, the code file 'main.c' is displayed with the following content:

```
1 #include <stdio.h>
2
3
4 void transpose(int a[2][2]) {
5     for(int i=0;i<2;i++) { for(int j=0;j<2;j++)
6         printf("%d ", a[j][i]); printf("\n");
7     }
8 }
9
10 int main() {
11     int a[2][2]={{1,2},{3,4}};
12     transpose(a); return 0;
13 }
14
15
16 }
```

At the top right, there are several icons: a copy icon, a share icon, a 'Run' button, and a 'Clear' button. The 'Run' button is highlighted in blue. To the right of the code area is the 'Output' pane, which shows the execution results:

```
1 3
2 4
*** Code Execution Successful ***
```

---

The End

---