Unit 5

Q1.Write a program to find the greatest of three numbers using function.
=>

```
#include <stdio.h>
int findMaximum(int num1, int num2, int num3)
{
   if (num1 >= num2 && num1 >= num3)
      return num1;
   else if (num2 >= num1 && num2 >= num3)
      return num2;
   else
      return num3;
}
int main()
{
   int num1, num2, num3;

   printf("Enter three numbers: ");
   scanf("%d %d %d", &num1, &num2, &num3);

   int max = findMaximum(num1, num2, num3);

   printf("The greatest number is: %d\n", max);

   return 0;
}
```

Output:

```
Enter three numbers: 1 2 3
The greatest number is: 3
```

Q2.What is function in C? Explain with example call by value function call.
=>

In C, a function is a block of code that performs a specific task and can be called from other parts of the program. Functions provide modularity and reusability in a program by allowing you to divide the code into smaller, manageable units.

When you call a function in C, you can pass arguments to the function. There are two ways to pass arguments to a function: "call by value" and "call by reference."

Call by Value Example:

In call by value, the actual values of the arguments are passed to the function. Any changes made to the parameters within the function do not affect the original values in the calling function.

```
#include <stdio.h>

// Function prototype
void addTen(int num);

int main() {
   int number = 5;

   printf("Original number: %d\n", number);

   // Calling the function with call by value
   addTen(number);

   // The original value remains unchanged
   printf("Number after function call: %d\n", number);

   return 0;
}

// Function definition
void addTen(int num) {
   num = num + 10;
   printf("Number inside function: %d\n", num);
}
```

In this example, the addTen function takes an integer num as a parameter. When you call this function from main, the value of number is passed to addTen. However, changes made to num inside the function do not affect the original number in the main function.

Output:

```
Original number: 5
Number inside function: 15
Number after function call: 5
```

As you can see, the original number remains 5 even after the addTen function is called because the function works with a copy of the value.

Q3.What is a function? How to declare, define and call a function.
=>

Function in C:

A function in C is a self-contained block of code designed to perform a specific task. It is a modular and reusable unit that helps in organizing and simplifying code. A function may take input parameters, perform operations, and return a value.

Declaration, Definition, and Calling a Function in C:

1. Declaration:
- Function declaration provides information about the function to the compiler, including its name, return type, and parameters (if any).
- It usually appears at the beginning of the program or in a header file.

```
// Function declaration
int addNumbers(int a, int b);
```

2. Definition:
- The function definition contains the actual implementation of the function.
- It specifies how the function behaves and what operations it performs.

```
// Function definition
int addNumbers(int a, int b) {
    return a + b;
}
```

3. Function Call:
- To use a function, you call it within another part of the program.
- Provide actual values (arguments) for the parameters declared in the function.

```
int main() {
    int result = addNumbers(5, 7); // Function call
    return 0;
}
```

Example:

```
#include <stdio.h>
int addNumbers(int a, int b);
int main()
{
    int result = addNumbers(5, 7);
    printf("Sum: %d\n", result);

    return 0;
}
int addNumbers(int a, int b)
{
    return a + b;
}
```

Q4.Write a program to swap the values of two variables using call by value function call mechanism.
=>

```
#include <stdio.h>
void swapValues(int a, int b)
{
   int temp = a;
   a = b;
   b = temp;
}
int main()
{
   int num1 = 5, num2 = 10;

   printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

   swapValues(num1, num2);

   printf("After swapping: num1 = %d, num2 = %d\n", num1, num2);

   return 0;
}
```

Output:
Before swapping: num1 = 5, num2 = 10
After swapping: num1 = 5, num2 = 10

Q5.What are the various advantages of using a function? Explain each?
=>
Certainly! Here are the key advantages of using functions in C:

1. *Modularity:* Functions break down a program into manageable modules.
2. *Code Reusability:* Functions can be reused in different parts of a program.
3. *Readability:* Functions improve code readability by focusing on specific tasks.
4. *Ease of Debugging:* Functions simplify debugging by isolating errors to specific parts.
5. *Abstraction:* Functions hide implementation details, providing a high-level view.
6. *Parameter Passing:* Functions allow data to be passed between different parts of a program.
7. *Namespaces:* Functions create their own namespaces, preventing naming conflicts.
8. *Facilitates Testing:* Functions can be tested individually, aiding in error identification.

Q6.Compare actual parameters and formal parameters.

=>

In C, actual parameters and formal parameters are terms related to function parameters. Let's understand the difference between them:

1. Actual Parameters:

- Definition:Actual parameters, also known as arguments, are the values or expressions supplied in the function call. They are the real values that are passed to a function when it is invoked.
- Role:Actual parameters provide the input values for the function during its execution.
- Example:

```
int result = addNumbers(5, 7);
 // 5 and 7 are actual parameters
```

2. Formal Parameters:

- Definition:Formal parameters are the parameters declared in the function definition. They act as placeholders for the values that will be passed to the function when it is called.
- Role: Formal parameters represent the variables used inside the function to receive and work with the values passed as actual parameters.
- Example:

```
int addNumbers(int a, int b)
{
    // a and b are formal parameters
    return a + b;
}
```

In summary, actual parameters are the real values passed to a function during its invocation, while formal parameters are the placeholders in the function definition that receive and work with these values. The values of actual parameters are assigned to the corresponding formal parameters during the function call.

Q7.Write a function program to read 10 integer element array and find the greatest of the 10 integers.
=>

```
#include <stdio.h>
int findGreatest(int arr[], int size)
{
   int greatest = arr[0];

   for (int i = 1; i < size; i++)
   {
      if (arr[i] > greatest)
      {
         greatest = arr[i];
      }
   }

   return greatest;
}

int main()
{
   int numbers[10];

   printf("Enter 10 integers:\n");
   for (int i = 0; i < 10; i++)
   {
      scanf("%d", &numbers[i]);
   }

   int greatest = findGreatest(numbers, 10);

   printf("The greatest element is: %d\n", greatest);

   return 0;
}
```

Output:

```
Enter 10 integers:
1 2 3 4 5 6 7 8 9 10
The greatest element is: 10
```

Q8. What is a function? How to declare, define and call a function

=>

Ref. Q.3