Answer Key for Function in C

Section A (MCQ Questions)

Q1: What is the correct syntax for declaring a function in C?
Answer: a) return_type function_name(parameters);

Q2: What is the purpose of the return statement in a function?
Answer: b) To return a value to the calling function

Q3: How many values can a function return using the return statement?
Answer: b) 1

Q4: In call by value:
Answer: a) The actual value is passed to the function

Q5: Every recursive function must have:
Answer: b) A base (termination) condition

Section B (Descriptive Questions)

Q6: Differentiate between library functions and user-defined functions with examples.

Basis	Library Functions	User-defined Functions
Definition	Predefined functions provided by C library.	Functions created by the user.
Header Files	Declared in header files like stdio.h, math.h.	Declared by the user in the program.
Example	printf(), scanf(), sqrt(), strlen()	sum(), area(), display()
Usage	Used directly by including the library.	Must be defined and called by the programmer.

```
#include <stdio.h>  // Required for printf
// User-defined function
int sum(int a, int b) {
    return a + b;
```

Answer Key for Function in C

```
}
int main() {
    // Using library function
    printf("Hello\n");
    // Using user-defined function
    int result = sum(5, 3);
    printf("Sum = %d\n", result);
    return 0;
}
```

OR

Feature	Library Functions	User-Defined Functions
Definition	Functions that are pre-written and provided by the C standard library.	Functions written by the programmer for a specific task.
Need to Write Code?	No, they are ready to use.	Yes, you have to write the function yourself.
Example	printf(), scanf(), sqrt(), strlen()	A function like int sum(int a, int b) to add two numbers.
Purpose	To save time and avoid rewriting common tasks.	To perform specific tasks not covered by library functions.
Flexibility	Fixed; cannot change their behavior.	Fully flexible; you can design it as needed.

Q7: Explain the process of function definition in C with a suitable example.

Answer:
A function definition tells the compiler what the function does.
The general syntax is:

```
#include <stdio.h>
int add(int a, int b) // function definition
{
   return a + b;
}
int main()
{
   int result = add(5, 3); // function call
```

```
    printf("Sum = %d", result);
    return 0;
}
```

OR

Function Definition in C

A function definition is where you actually write the code for the function—the steps it will perform. It tells the computer:

- What the function does
- What inputs it takes (parameters)
- What it returns (output)

A function has four main parts:

1. Return Type – What type of value the function will give back (like int, float, void).
2. Function Name – Name you give to the function.
3. Parameters (Optional) – Values you pass to the function.
4. Function Body – The code that executes when the function is called.

```
#include <stdio.h>
// Function Definition
int square(int n) {
    return n * n;
}
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    printf("Square of %d is %d\n", num, square(num));
    return 0;
}
```

Q8: Differentiate between call by value and call by reference with examples.

Call by Value	Call by Reference
A copy of the actual value is passed.	The address of the variable is passed.
Changes in the function do not affect the original variable.	Changes affect the original variable.

Answer Key for Function in C

Call by Value	Call by Reference
Default in C.	Implemented using pointers.

Example (Call by Value):
```
void change(int x) {
    x = 10;
}
```
Example (Call by Reference):
```
void change(int *x) {
    *x = 10;
}
```

OR

Feature	Call by Value	Call by Reference
Meaning	Copies the value of the argument into the function parameter.	Passes the address of the variable to the function.
Effect on Original Variable	Original variable does NOT change.	Original variable can be changed inside the function.
How it is done in C	Just pass the variable directly.	Pass the pointer (address) of the variable using &.
Memory Used	Uses extra memory for the copy.	No extra copy; function works on original variable.
Example	int a = 5; fun(a);	int a = 5; fun(&a);

Q9: Explain the concept of function arguments and parameters with examples.

Answer:

- **Parameters** are variables defined in the function header that receive values.
- **Arguments** are actual values passed when the function is called.

```
#include <stdio.h>  // Required for printf
```

```
// Function to calculate the sum of two integers
int sum(int a, int b) {
    return a + b;
}
int main() {
    int result = sum(5, 10);  // 5 and 10 are arguments
    printf("%d\n", result);   // Output the result
    return 0;
}
```

OR

Parameters in C

- Parameters are variables listed in the function definition.
- They act as placeholders for the values the function will use.

Arguments in C

- Arguments are the actual values passed to the function when calling it.

```
#include <stdio.h>
void greet(char name[]) {  // 'name' is a parameter
    printf("Hello, %s\n", name);
}
int main() {
    greet("Alice");  // calling function with an argument
    return 0;
}
```

Ques 10: What is a base condition in recursion? Why is it necessary?

Answer:

A base condition (or base case) in recursion is the condition under which a recursive function stops calling itself.

Why it is necessary:

- Without a base condition, the recursion will continue indefinitely and cause a stack overflow.
- It provides a termination point for the recursive calls.
- It ensures that the problem is eventually solved by breaking it down into smaller sub-problems.

OR

Answer Key for Function in C

- Base condition is the rule in a recursive function that tells it when to stop.
- It prevents infinite recursion (so the program doesn't crash).
- It provides a direct answer for the simplest case.
- Without it, the function would keep calling itself forever.

Why it is necessary:

- To prevent infinite recursion (so the program doesn't crash).
- To provide a simple answer directly, which helps build the final solution.

Section C (Practical Questions)

Ques 11(a): Program to calculate the area of a circle using a user-defined function

```
#include <stdio.h>
#define PI 3.14159
// Function to calculate the area of a circle
float areaOfCircle(float radius) {
    return PI * radius * radius;
}
int main() {
    float radius;
    printf("Enter the radius of the circle: ");
    scanf("%f", &radius);
    float area = areaOfCircle(radius);
    printf("Area of the circle: %.2f\n", area);
    return 0;
}
```

Ques 11(b): Recursive function to find GCD of two numbers.

```
#include <stdio.h>
// Recursive function to find GCD
int gcd(int a, int b)
{
    if (b == 0)  // Base condition
    {
        return a;
    }
    else
    {
        return gcd(b, a % b);
```

Answer Key for Function in C

```
    }
}

int main()
{
    int num1, num2;
    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);
    int result = gcd(num1, num2);
    printf("GCD of %d and %d is %d\n", num1, num2, result);

    return 0;
}
```