

BUPT SE ICP Quiz -- Unit 05 Functions and Libraries

If you are using LLM to help you in this quiz, you MUST write down 1) the name of the LLM and 2) the PROMPTS. Otherwise, you will get zero score if the instructor finds that you are using LLM.

You may use the following table as a template, putting the table down to the title of each question.

| Name of LLM | Prompts |
|-------------|---------|
| | |

- The prototype of the function sqrt in the math library is `double sqrt(double x);`
- [T] T/F: There can be more than one return statement in the body of a function.
- The math library contains a function with the following prototype:
`double atan2 (double, double);`
 Even if you have no idea what this function does, what information does the prototype give you about using this function?

 Using this function needs 2 variables both of which should be stated with “double” statement. And will return a value which is also a double float num.
- Use LLM to help you answer: What is a predicate function?

| Name of LLM | Prompts |
|-------------|---|
| LongCat AI | 1. I'm a beginner in learning CS and I want to know about some questions. 2. What is a predicate function? Plz give a brief introduction. 3. All of the examples ought to be written in C (C99) language. |

AI Answer:

What is a Predicate Function?

A predicate function (also called a predicate) is a function that:

Takes one or more input arguments

Returns a boolean result (true or false)

Typically used to test whether certain conditions hold for the input(s)

In C99, since there is no built-in boolean type in traditional C, we use:

stdbool.h header for bool, true, and false (C99 standard)

Return type bool instead of int (though both work)

- Use LLM to help you answer: What is the relationship between arguments and formal parameters?

| Name of LLM | Prompts |
|-------------|--|
| LongCat AI | I'm a beginner in learning CS and I want to know about some questions. What is the relationship between arguments and formal parameters? Code in C |

Core Principles

1. One-to-One Mapping

- Each argument corresponds to a formal parameter by position (order matters!)

- Number of arguments must match number of formal parameters (except for variadic functions)

2. Value Passing Mechanism

- C uses **pass-by-value** by default (copies the value of the argument to the parameter)

3. Scope

- Formal parameters are local variables within the function

6. Answer with LLM: Variables declared within a function are said to be local variables. What is the significance of the word local in this context?

| Name of LLM | Prompts |
|-------------|--|
| LongCat AI | 1. I'm a beginner in learning CS and I want to know about some questions. 2. Variables declared within a function are said to be local variables. What is the significance of the word local in this context? Plz give a brief introduction. 3. All of the examples ought to be written in C (C99) language. |

Scope Limitation

The word local in this context means the variable:

Is accessible only within the function where it is declared

Cannot be directly accessed by any other function or code block- Has lexical scope restricted to its enclosing function

Practical Benefits

Modularity

Functions become self-contained building blocks that don't interfere with each other.

Data Hiding

Sensitive/implementation details are hidden from the outside world (encapsulation).

Memory Efficiency

Stack space is recycled after function execution (reduces memory footprint).

Debugging Safety

Harder to accidentally overwrite variables in unrelated parts of the program.

Parallel Development

Multiple programmers can work on different functions without naming collisions.

Advanced Note: Storage Duration

Local variables are also called automatic variables because:

They are created automatically when the function is called

They are destroyed automatically when the function exits- This contrasts with static variables (which persist across calls) or global variables (program lifetime)

7. Write a function `isPrime()` to determine if a number is prime. Call the function in `main()` to check if 101 is prime.

```
#include<stdio.h>
#include<math.h>
```

```
int isPrime(int i){
    for (int j = 1; j <= sqrt(i); j++){
        if (i % j == 0){
            return 0;
        }
        return 1;
    }
}
```

```
int main(void){
    int testNumber = 23;
    if(isPrime(testNumber)==1){
        printf("101 is a Prime\n");
    }
    else{
        printf("101 is not a Prime\n");
    }
    return 0;
}
```

8. Write a function `circle()` to calculate the circumference and area of a circle. Call separate functions to calculate circumference and area within `circle()`. Call `circle()` in `main()` to output the circumference and area of a circle with radius 5.

```
#include<stdio.h>
#define pi 3.14
```

```
double circle(int r, char c){
    if (c == 'C'){
        printf("Circumstance is %.2f", 2*pi*r);
    }
    else if(c == 'S'){
        printf("S is %.2f", pi*r*r);
    }
    else{
        printf("Wrong Input\n");
    }
}
```

```
}
```

```
int main(void){  
    int r = 5;  
    char what;  
    printf("Plz input C/S to calculate\n");  
    scanf("%c", &what);  
    circle(r, what);  
    return 0;  
}
```

9. Define a global variable sum, write function increment() to add 1 to it, and function getSum() to return the value of sum. Call the functions in main() to verify.

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

```
int sum = 0;
```

```
void increment() {  
    sum += 1;  
}
```

```
int getSum() {  
    return sum;  
}
```

```
int main() {  
    increment();  
    increment();  
    printf("Sum: %d\n", getSum());  
    return 0;  
}
```

10. In file1.c, define an external variable count. Call this external variable in the main() function of file2.c.

File1.c

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

```
int n = 42;
```

```
int main() {  
    printf("Love CS\n");  
    return 0;  
}
```

```
}
```

File2.c

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

```
int main(){  
    extern int n;  
    printf("External variable is %d\n", n);  
    return 0;  
}
```

11. Write a recursive function factorial to calculate the factorial of a number. Call the function in main() to calculate the factorial of 5.

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

```
int factorial(int n) {  
    for(int i = n - 1; i > 1; i--) {  
        n *= i;  
    }  
    return n;  
}
```

```
int main() {  
    int num = 5;  
    int result = factorial(num);  
    printf("Factorial of %d is %d\n", num, result);  
    return 0;  
}
```

以下题目为选做，不计入作业考核。以下题目均需用到“字符数组”的知识

12. Write a function letterCount() to count the number of letters in a string. Call the function in main() to count letters in "Hello World".

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

```
int letterCount(char str[]){  
    int count = 0;  
    for (int i = 0; str[i] != '\0'; i++){  
        if (str[i] != ' '){  
            count += 1;  
        }  
    }  
}
```

```
}  
return count;  
}  
  
int main(){  
    int countNum;  
    char testStr[15] = "Hello World";  
    countNum = letterCount(testStr);  
    printf("There's %d num in Hello World\n", countNum);  
    return 0;  
}
```

13. Write a function equalString() to check if two strings are equal. Call the function in main() to check if "abc" and "abc" are equal.

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

```
int equalString(char str1[], char str2[]) {  
    int i;  
    for (i = 0; str1[i] != '\0' && str2[i] != '\0'; i++) {  
        if (str1[i] != str2[i]) {  
            return 0;  
        }  
    }  
    return (str1[i] == '\0' && str2[i] == '\0'); //AI improved this sentence cuz mine is a  
    shitcode  
}
```

```
int main() {  
    char testStr1[] = "abc", testStr2[] = "abc";  
    if (equalString(testStr1, testStr2) == 1) {  
        printf("These 2 strings are equal\n");  
    } else {  
        printf("These 2 strings are not equal\n");  
    }  
    return 0;  
}
```