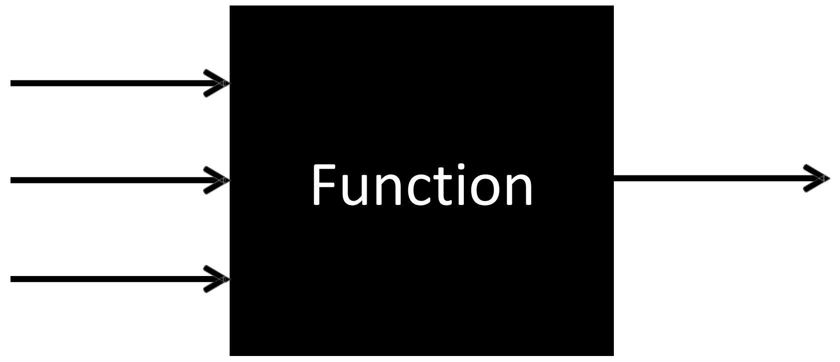
C PROGRAMING

Ketan Kore

Sunbeam Infotech

Functions

- C program is made up of one or more functions.
- Programs are divided into multiple logical parts called as function or sub-routine
- C program contains at least one function i.e. main() function.
  - Execution of C program begins from main.
  - It returns exit status to the system.
- Advantages
  - Reusability
  - Readability
  - Maintainability



- Function is set of instructions, that takes zero or more inputs (arguments) and return result (optional).
- Function is a black box.

Functions

- Each function has
  - Declaration
  - Definition
  - Call
- A function can be called one or more times.
- Arguments
  - Arguments passed to function → Actual arguments
  - Arguments collected in function → Formal arguments
  - Formal arguments must match with actual arguments

Examples:

1. addition()
2. print_line()
3. factorial()
4. combination()

Functions

- Function Declaration
  - Informs compiler about function name, argument types and return type.
  - Usually written at the beginning of program (source file).
  - Can also be written at start of calling function).
  - Examples:
    - **float divide(int x, int y);**
    - **int fun2(int, int);**
    - **int fun3();**
    - **double fun4(void);**
    - **void fun5(double);**
  - Declaration statements are not executed at runtime.

- Function Definition
  - Implementation of function.
  - Function is set of C statements.
  - It process inputs (arguments) and produce output (return value).

```
float divide(int a, int b) {
        return (float)a/b;
}
```

  - Function can return max one value.
  - Function cannot be defined in another function.
- Function Call
  - Typically function is called from other function one or more times.

Function execution

- When a function is called, function activation record/stack frame is created on stack of current process.
- When function is completed, function activation record is destroyed.
- Function activation record contains:
  - Local variables
  - Formal arguments
  - Return address
- Upon completion, next instruction after function call continue to execute.

Function types

- User defined functions
  - Declared by programmer
  - Defined by programmer
  - Called by programmer
- Library (pre-defined) functions
  - Declared in standard header files e.g. stdio.h, string.h, math.h, …
  - Defined in standard libraries e.g. libc.so, libm.so, …
  - Called by programmer
- main()
  - Entry point function – code perspective
  - User defined
  - System declared
  - int main(void) {…}
  - int main(int argc, char *argv[]) {…}

Storage class

	Storage	Initial value	Life	Scope
auto / local	Stack	Garbage	Block	Block
register	CPU register	Garbage	Block	Block
static	Data section	Zero	Program	Limited
extern / global	Data section	Zero	Program	Program

- Each running process have following sections:
  - Text
  - Data
  - Heap
  - Stack
- Storage class decides
  - Storage (section)
  - Life (existence)
  - Scope (visibility)
- Accessing variable outside the scope raise compiler error.

Storage class

- Local variables declared inside the function.
  - Created when function is called and destroyed when function is completed.

- Global variables declared outside the function.
  - Available through out the execution of program.
  - Declared using extern keyword, if not declared within scope.

- Static variables are same as global with limited scope.
  - If declared within block, limited to block scope.
  - If declared outside function, limited to file scope.

- Register is similar to local storage class, but stored in CPU register for faster access.
  - register keyword is request to the system, which will be accepted if CPU register is available.

Thank you!

Ketan Kore<Ketan.Kore@sunbeaminfo.com>