Chapter 07 – C Programming – Part I

Introduction

- C is the successor to B
- C is a compiled, procedural language
- Very powerful (thanks to pointers)
- Systems programming mostly in C
- Most kernels/ low-level programs in C
- The Linux kernel is written in C

First Program

- Create a file sample.c
- #include <stdio.h>

```
int main ( ){
    printf (“Hello World”);
    return 0;
}
```

Compiling

- The .c file must be compiled into an object file
- Various compilers exist to do this conversion
- We shall use the popular GNU C Compiler
- From the terminal,

  gcc -o *objectfilename sourcefilename.c*
- Example:

  gcc -o sample sample.c

Running

- Once compiled, the generated object file can be run as an executable
- Example, from the same directory

  ./sample
- You do not need to change permissions, since the compiler makes it executable

Variables

- Similar to C++
- Data types int, float, double, long, char, etc.
- Variables must be declared, initialized, and then used
- Example

```
int x = 7;
int y;
y = x + 10;
```

Operations

- Similar to C++
- +, -, /, *, %, ++, --
- +=, -=, /=, *=, etc.
- ==, >=, <=, !=, >, <
- &&, ||, !
- &, |, ^, ~, <<, >>

Input/ Output

- From the library stdio.h
- Output
  printf ( )
- Input
  scanf ( )

Output

- printf separates formatting from variable data
- So, to print “The value of variable is x”

  printf (“The value of variable is %d” , x)
- Notice, the string contains ‘placeholders’ (format specifier) where you want the values of the variables
- The variables are then given as additional arguments to printf( )
- The ‘placeholders’ specify the data type of the variable, e.g. %d is for integers

Some Format Specifiers

- Format Specifier: Type
- %c Character
- %d Signed integer
- %f Float values
- %l or %ld or %li Long
- %lf Double
- %o Octal representation
- %p Pointer
- %s String
- %u Unsigned int
- %x or %X Hexadecimal representation

Input

- The function scanf ( ) waits for input from the keyboard, and stores the data in a variable
- Example

```
int x;
scanf ( “%d”, &x );
```

- Notice the format specifier and passing the reference of the variable (&x, NOT x).

Input

- The function scanf ( ) waits for input from the keyboard, and stores the data in a variable
- Example

```
int x;
scanf ( “%d”, &x );
```

- Notice the format specifier and passing the reference of the variable (&x, NOT x).

Conditional Statements

- Similar to C++
- if (condition){
  statements to execute
  }
  else{
  statements for else
  }
- switch (expression){
  case val1:
  case val2:
  :
  }

Loops

- Similar to C++
- while (condition) {
  iteration_body
  }

- do{
  iteration_body
  }while (condition)

- for (var=init; condition; increment/decrement){
  iteration_body
  }

Functions

- Similar to C++

- return_type function_name(parameter list) {
  body of the function
  }

- Example:

```
int add ( int x, int y ){
    int z;
    z = x + y;
    return z;
}
```