ME 172

C Programming Language Sessional

Lecture 3

Introduction to C Program

- C is a Structured Programming

Basic Structure of a C Program

```
# Header File
        #include<stdio.h>
# Define CONSTANTS
# Function Prototype Declaration
```

Main Function Declaration

int main()
{
return 0;
}
	void
	{
	
	}

```
int main()
{
variable declaration;                    /* Every variable used in the function must be declared locally or globally */
        int a=5; float x = 7.0;
Library function
scanf("format", & variable);             // input function (comment)
        scanf(%d %f",&a,&x);
User defined functions
if/for loop/D0-while loop;               //control/logical statements
printf("format", variable/expression);   // output statements
        printf("%d %f",a,x);
return  0;
}
```

Introduction to C Program

- C is a Structured Programming

Basic Structure of a C Program

```
# Header File
Example:  #include<stdio.h>
# Define CONSTANTS
# Function Prototype Declaration

                                        Main Function Declaration

int main()
{
variable declaration;                   /* Every variable used in the function must be declared locally or globally */
Example: int a=5; float x = 7.0;
Library function
scanf("format", & variable);            // input function (comment)
Example: scanf(%d %f",&a,&x);
User defined functions
if/for loop/DO-while loop;              //control/logical statements
printf("format", variable/expression);  // output statements
Example: printf("%d %f",a,x);
return  0;
}
```

variables
scanf()
printf()

Data Types and Modifier

- Basic data types are
  - *char*
  - *int*
  - *float*
  - *double*
- Modifiers
  - signed
  - unsigned
  - short
  - long

Data Types and Modifier

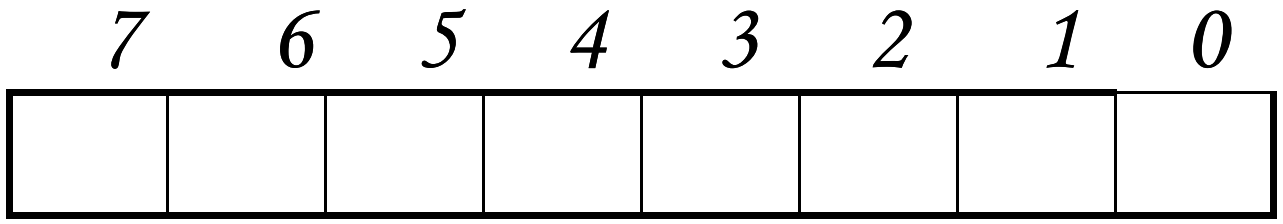
Type	Size (bytes)
`long`	4
`short`	2
`char`	1
`int`	machine-dependent
`float`	4
`double`	8
`void`	no size

Bit and Byte

- Each piece of information stored within computer's memory is encoded as some unique combination of zero and ones.

- These 0/1 are called bits.

1 byte = 8 bits.



Variables

- All variables must be declared before they use.
- There are two places variables are declared
  - Variable declared outside all functions called ***Global Variable***, they can be accessed by any function in the program
  - Variable declared inside a function called ***Local Variable,*** that can only be accessed by only the function in which it is declared
- Variables are declared as follows:

  type-name variable-name;
- Example - *int a;*
- Where variables are declared?

Variables

- Rules of declaring variables
  - Alphabetic character (a…….z ;A…..Z) , digits (0,1……9), (_) and ($)can only be used in variable name. (*int number*)
  - 1st character must be letter, cannot be digit. (*1_roll*)
  - Both upper and lowercase are permitted
  - No space is allowed in the variable name (*my name*)
  - Keywords are not allowed (*void, int, float etc.*)
  - Variable name should not be greater than 31 char.
  - Variables have unique names

Variable Use

- Variables must be set (initialized) before they can be read
- riables are assigned values by use of the "="

```
int   x;       // Variable Declaration
x=5;           // Variable Initialization
```

Or

```
int x=5;
```

Format Specifier

%d	signed decimal Integer
%u	unsigned decimal integer
%ld	long integer
%f	floating point data type
%lf	double data type
%Lf	long double
%e	float data in exponential e notation
%c	single Character
%s	string pointer ,Prints characters until a null-terminator is pressed.
%%	prints the % character

scanf() function

- *scanf* function allows to accept input from standard in, generally the keyboard

➢ General form

  - scanf("format_specifier",&variable);

  - "&variable" means address of the variable

➢ *int age;*

scanf ("%d", &age);

scanf() function

- More example
- *float gpa;*

  scanf ("%f", &gpa);
- *char grade;*

  scanf ("%c", &grade);
- *double number;*

  scanf(%ld", &number);

scanf() function

- More examples
- ```
  #include<stdio.h>
  void main()
   {
   int num, float  f;
    scanf ("%d", &num);
   scanf ("%f", &f);
    }
  ```

} scanf ("%d %f", &num, &f);
  ```

printf() function

- The *printf* statement allows to send output to standard out; standard out is generally the screen

- printf general form

  printf("format specifier", variable);

printf() function

- Examples
- *#include<stdio.h>*

```
void main()
{
int x = 10;
printf("%d", x);
printf("The value of x is %d", x);
}
```

- *Output is*

10 The value of x is 10

Escape Sequences

Escape Sequence	Character Value
'\b'	Blank space
'\n'	New line
'\t'	Tab
'\\'	Backslash
'\''	Apostrophe
'\"'	Double quote

Formatted Output

Output of Integer Numbers						% wd
Format	Output					
printf("%d", 9876);	9	8	7	6		
printf("%6d", 9876);	00		9	8	7	6
printf("%2d", 9876);	9	8	7	6		
printf("%-6d", 9876);	9	8	7	6		
printf("%06d", 9876);	0	0	9	8	7	6

Formatted Output

Output of Real Numbers	% w.p f	% w.p e										
Format (y = 98.7654)	Output											
printf("%7.4f", y);	9	8	.	7	6	5	4					
printf("%7.2f", y);			9	8	.	7	7					
printf("%-7.2f", y);	9	8	.	7	7							
printf("%f", y);	9	8	.	7	6	5	4					
printf("%10.2e", y);			9	.	8	8	e	+	0	1		
printf("%11.4e", -y);	-	9	.	8	7	6	5	e	+	0	1	
printf("%-10.2e", y);	9	.	8	8	e	+	0	1				
printf("%e", y);	9	.	8	7	6	5	4	0	e	+	0	1

math.h

Some functions of math.h

- pow(b,e)
- sin(x)
- cos(x)
- tan(x)
- log(x)
- log10(x)
- abs(x)

and many others

```
#include<stdio.h>
#include<math.h>
#define pi 3.141592
int main()
{
        double sum;

        sum=cos(pi);
        printf("%lf",sum);
        return 0;

}
```

That’s all about today….