Programming Notes

N.R.Aravind

I.I.T. Hyderabad

Topics for review

- Basics of a C program
- int and float variables: Input and Output
- if...else: logical operators
- char variable
- Extras

Basic outline of a C program

A simple C program: helloWorld.c

```
#include<stdio.h>
int main()
{
    printf("Hello, World");
return 0; }
```

A simple C program: helloWorld.c

```
#include<stdio.h>
// stdio.h: Standard input-output header file
// Contains declaration of printf
int main() // Main point of execution
{
// Code goes here.
    printf("Hello, World");
return 0;
}
```

Compiling and running a C program

$ gcc helloWorld.c -o hello

- gcc: Gnu C Compiler
- Translates the C program into machine code named "hello"
- -o: specifies the output file name

$ ./hello

- Run (execute) the program named "hello"
- To run a file named "xyz", type ./xyz (Linux) and xyz (Windows)

Sequential execution

```
int a=10, b=20;
      a=a*b;
      b=a-b;
// a=? b=?
```

Sequential execution

```
int a=10, b=20;
      a=a*b;
      b=a-b;
// a=? b=?
// a=200 b=180
```

Sequential execution

```
int a=10, b=20;
      b=a-b;
      a=a*b;
// a=? b=?
```

Sequential execution

```
int a=10, b=20;
      b=a-b;
      a=a*b;
// a=? b=?
// a=100 b=10
```

Sequential execution

```
int a;
printf("\n The value is %d",a);
a=5;
printf("\n The value is %d",a);
```

Points to note

- C statements usually end in a semicolon.
- printf $\neq$ Printf. CASE-sensitive.
- Variable names: avoid keywords.
- Use // for single-line comment.
- Use /* Comments */ for multi-line comments.

int and float variables

float

- float num1=2.16789;
- printf("%f",num1); // Prints 2.167890
- printf("%.2f ", num1);

float

- float num1=2.16789;
- printf("%f",num1); // Prints 2.167890
- printf("%.2f ", num1);   // Prints 2.17

float

- float num1=2.16789;
- printf("%f",num1); // Prints 2.167890
- printf("%.2f ", num1);   // Prints 2.17
- printf("%.4f", num1);

float

- float num1=2.16789;
- printf("%f",num1); // Prints 2.167890
- printf("%.2f ", num1);   // Prints 2.17
- printf("%.4f", num1);   //Prints 2.1679

Input

```
int x; float y;
printf(”Enter a value for x: ”);
scanf(”%d”,&x);
printf(”Enter a value for y: ”);
scanf(”%f”,&x);
```

if…else

if (__) {...}

```
if (num < 0)
{
  num=-num;
}
 printf(" %d",num);
```

The if ... else statement

Syntax:

```
    if (expression)
    {
    Statements S1
    }
    else
    {
    Statements S2
    }
//If the expression is true, S1 will be executed,
otherwise S2 will be executed.
```

The NOT operator

```
if (!(num == 0))
{
 printf(" It's non-zero!");
}
```

The AND operator

if $((num >= 1) \&\& (num <= 100))$

The OR operator

if $((num < 1) || (num > 100))$

char variables

Example 1

```
char ch;
ch='A';
printf("Enter a character: ");
ch=getchar();
```

Example 2

```
char answer;
int score;
printf("What is the capital of Latvia?");
printf("a. Tallinn");
printf("b. Riga");
printf("c. Minsk");
printf("d. Warsaw");
printf("Enter your choice: ");
answer=getchar();
```

Extras

Variables in memory

Address	Value
2300	17
2301	255
2302	35
2303	6
2304	29
2305	194
2306	.
2307	.
2308	.

```
int a,b,c;
```

Variables in Memory

	Address	Value
	2300	17
a	2301	255
	2302	35
	2303	6
b	2304	29
	2305	194
	2306	.
c	2307	.
	2308	.

```
int a,b,c;
```

Memory

2300	0	0	0	1	0	0	0	1
2301	1	1	1	1	1	1	1	1
2302	0	0	1	0	0	0	1	1
2303	0	0	0	0	0	1	1	0
2304	0	0	0	1	1	1	0	1
2305	1	1	0	0	0	0	1	0
2306				.	.	.		
2307				.	.	.		
2308				.	.	.		

Addresses

```
int x;
printf("%p",&x);
```

TO BE CONTINUED...