MA 511: Computer Programming

Lecture 3

http://www.iitg.ernet.in/psm/indexing_ma511/y08/index.html

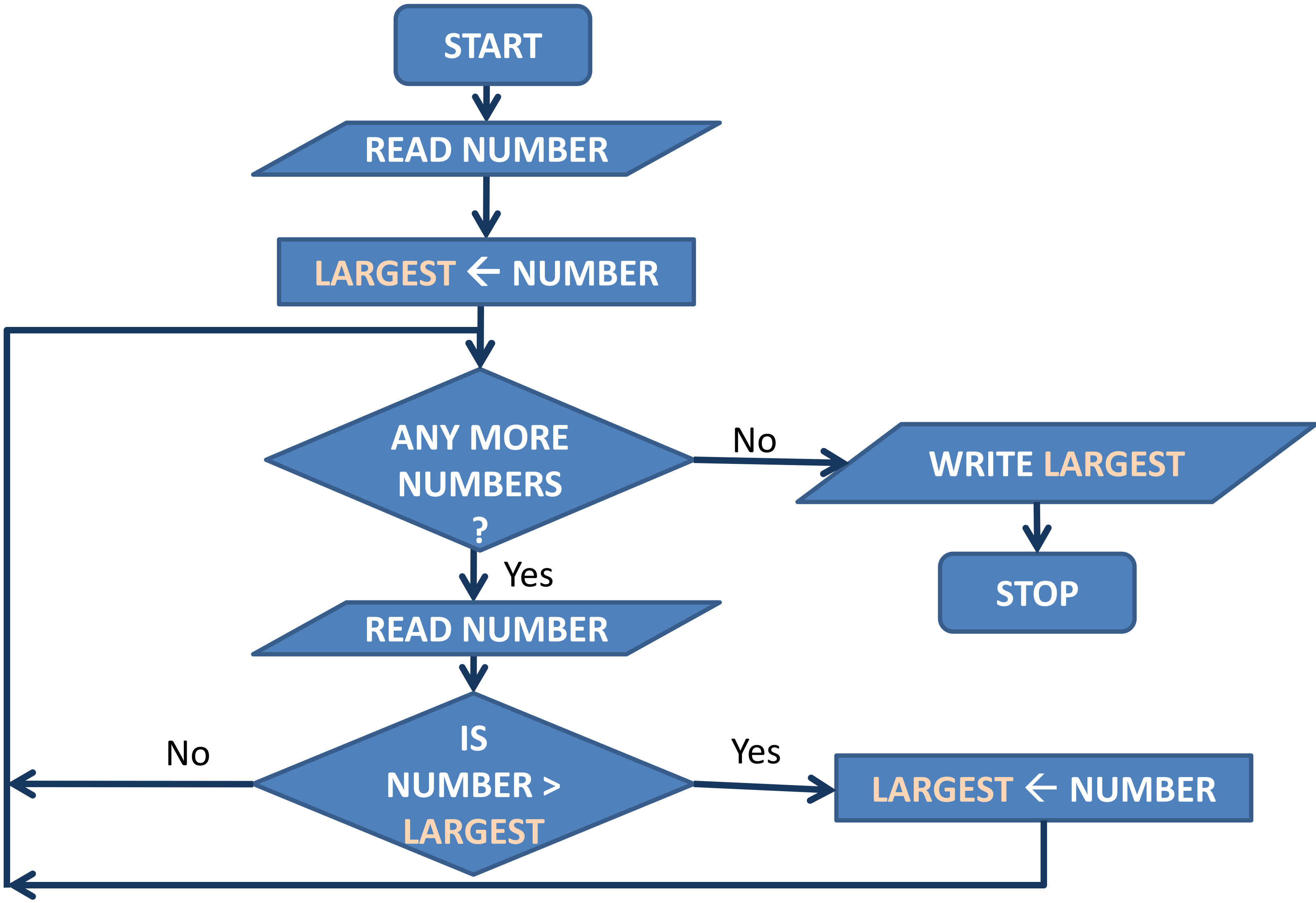
Partha Sarathi Mandal

psm@iitg.ernet.ac.in

Dept. of Mathematics, IIT Guwahati

Semester 1, 2008-09
Mon 10:00-10:55 Tue 11:00-11:55 Fri 9:00-9:55 Class: 1G2
MA512 Lab : Wed 14:00-16:55

Largest of the set of numbers



Exercise

1. Celsius to Fahrenheit conversation.
2. The largest of a set of numbers.
3. Given integer is prime or composite.
4. LCM of two given integers.
5. GCD of two given integers.
6. Solve: Ax^2 + Bx + C = 0
7. The Fibonacci numbers:

   Let $a_0 = 1$ and $a_1 = 1$, and an be given by the following recursive definition: $a_n = a_{n-1} + a_{n-2}$

Exercise

8. Write a program which takes in a positive integer and prints one factorization of it into primes.

Example C Program

```
/* Calculate the area of a circle */
#include<stdio.h>
main(){
        float radius, area;
        printf("Radius = ?  ");
        scanf("%f", &radius);
        area = 3.14159 * radius * radius;
        printf("Area = %f", area);
}  /* end of main */
```

Preprocessor directive, stdio.h is included in the compiled machine code at # . It contains the standard I/O routines. Must be in the first column. Must not end with a semicolon.

main() is a function, is required in all C pgs, it indicate start of a C pgs., main() it is not followed by a comma or semicolon.

Braces { and } enclosed the computations carried out by main()

Every statement is terminated by a **semicolon**

Declaration : it informs the compiler that **radius** and **area** are variables names and that individual boxes must be reserved for them in the memory of the computer.

Comments (remarks) are placed anywhere within the program within delimiters /* end of main */ or // end…

Code for **turbo C**

```
/* Calculate the area of a circle */
#include<stdio.h>
#include<conio.h>
main(){
        float radius, area;
        printf("Radius = ?  ");
        scanf("%f", &radius);
        area = 3.14159 * radius * radius;
        printf("Area = %f", area);
        getch();
}   /* end of main */
```

Example C Program

```
/* Calculate the area of a circle */
#include<stdio.h>
#define PI 3.14159
main(){
        float radius, area;
        int i, n;
        printf("n = ? ");
        scanf("%d", &n);
        for(i=1; i<= n; i=i+1){
                printf("Radius = ?  ");
                scanf("%f", &radius);
                area = PI * radius * radius;
                printf("Area = %f\n", area);
        } /* end of for */
} /* end of main */
```

Symbolic constant is a name that substitutes for a sequence of characters; numeric, character or string constant. It is replaced by its corresponding character constant during compile

printf & **scanf** are not part of the C language; there is no input or output defined in C itself. Its just a useful function from the standard library of functions that are normally accessible to C pgs. The behavior of printf & scanf are defined in the ANSI standard.

for is looping statement , the 1st expression specifies an initial value for an index, 2nd determines whether or not the loop is continued, 3rd allows the index to be modified at the end of each pass.

Datatytes

1. **Integer constant**

    int a, b; (for 32bit machine)

    long int a, b;

    Signed (- 2^{31} to + 2^{31} -1), Unsigned : (0 to 2^{32} -1)

    short int a, b;

    Signed: Considered 16 bit integer with range (- 2^{15} = -32768 to + 2^{15} -1 = +32767), Unsigned (0 to (2^{16} -1)=65535)

2. **Floating point constant**

    float a = 2.0, b = 0.9999, sum = 0.; [restricted within 3.4 x 10^{-38} to 3.4 x 10^{38}]

    double fact = 0.11236E-6; [0. 11236 x 10^{-6}] [within 1.7 x 10^{-308} to 1.7 x 10^{308}]

3. **Character constant**

    char a = ‘x’, b = ‘3’, c = ‘#’, text[18] = “kolkata”;

4. **String constant**

    string a = “Delhi, 100011”, b = “$20.95”;

Variable declaration

```
int i,j,k;
    long p,q;
    short s;
    unsigned u;
float r;
    double dr;
    long double lr;
char c, text[10];
```

Operators

Operator	purpose
+	addition
-	subtraction
*	multiplication
/	division
%	remainder after integer division (module operator)

Operands conversion

- float @ double = double @ : operator
- float @ long double = long double
- float @ char/long int/short int/int = float
- long int @char/short int/int = long int

  int i = 7;

  float f = 5.5;

  char c = ‘w’ [w= 119 (ASCII) American Standard Code for Information Interchange]

i+f	**= 12.5**	float
i+c	**= 126**	int
i+c-’0’	**= 78**	int [zero, 0= 48 (ASCII)]
(i+c)-(2*f/5)	**= 123.8**	float