INTRODUCTION TO IMPORTANT CONCEPTS OF MATHEMATICA

By

Dr Kumari Priyanka

Ms Deepti

WHILE LOOP

- It is used when we do not know the number of operations to be performed in advance.
- Number of iterations depends on some logical (relational) conditions (statement).
- Syntax

  While[test, body] , This will evaluate the body repeatedly until test first fails to return True

  While[test] , This run the loop with null (empty) body until test first fails to return True.
- **Example**: Print first five natural numbers using While loop

  Mathematica Code:

```
n=1;
While[n <= 5, Print[“n = ”, n]; n++]
n = 1
n = 2
n = 3
n = 4
n = 5
```

- Notice *n++* (increment operator), which increase the value of n by 1. Same way n--, decrease the value of n by 1.

Example: Find out sum of first 10 natural numbers using while loop.
Mathematica Code:

```
n = 10;
sumn = 0;
While[n >= 0, sumn = sumn +n; n--];
Print[“sum = “, sumn];
sum = 55
```

LIST

- ✓ A list is simply a set of elements. A list could be one dimensional, two dimensional or three dimensional and so on.
- ✓ Syntax:

  List[a, b, c, d, . . .], defines a list with elements a, b, c, d, ...

  Example: Define a list of first four even numbers.

  Mathematica Code:

  ```
  list1=List[2, 4, 6, 8]
  {2, 4, 6, 8}
  ```
- ✓ Other simple way to define a list is by enclosing the element in curly brackets.

  ```
  list2=[1, 2, x, x^2}
  ```
- ✓ The kth element of list1 can be accessed by **list1[[k]],** For example: the 3rd element of list1 is accessed by

  ```
  list1[ [3] ]
  6
  ```

  The 2nd element of list2 is accessed by:

  ```
  list2[ [2] ]
  2
  ```
- ✓ The above defined list can be treated as a vector or one dimensional array and many vector operations can be performed on it.

FOR LOOP

- ❑ Syntax

  For[start, test, incr, body] executes start, then repeatedly evaluates body and incr until test fails to give True.

- ❑ **Example:** Print first 4 non-negative integers using For loop.

  Mathematica Code:

```
For[i = 0, i < 4, i++, Print[ i ] ]
0
1
2
3
```

- ❑ **Break[ ]** breaks out of For loop
- ❑ **Example:** Define a list {3, -4, 6, -1, 7, -8} and square the first four elements.

  Mathematica Code:

```
list = {3, -, 6, -1, 7, -8};
n = Length[list];
For [ i =1, i<= n, i++, If[ i >4, Break [ ] ]; list [ [i] ] = list[ [i] ]^2];
Print["Updated list = ", list];
Updated list = { 9, 16, 36, 1, 7, -8}
```

- **Continue[ ]** can be used to skip execution of body part, if a certain condition is fulfilled.

- Example: Define a list {3, -4, 6, -1, -8, 9, 5} and add one to all the even elements of the list.
  Mathematica Code:

```
list = {3, -4, 6, -1, 7, -8 9, 5};
n = Length[list];
For[i=1, i<=n, i++, If [Mod[list[ [i] ], 2]≠ 0, Continue[ ] ]; list [ [i]]= list[ [i]]+1];
Print["Updated list = ", list];

Updated list = {3, -3, 7, -1, 7, -7, 9, 5}
```

- Notice that the statement **list [[i]]= list[[i]]+1** , is executed when the ith element **list[[i]]** is divisible by 2.
- Break[ ]: Exit from the evaluation of the body of loop or simply break the loop. This is used when we want to come out of the loop on satisfying certain condition.
- Return[expr]: Return the value expr from a function. This is used generally in the end of a function to return the results computed inside the function. The value of expr can be Null.

MODULE

- Syntax

  Module[{x, y, . . . }, expr], specifies that occurrences of the symbols x, y, . . . in expr should be treated as local

  Module[{x = x_0, ...}, expr], defines initial values for x, . . .

- Whenever we want to pass some variables to a function and in case we want that the variable remain safe (i.e., the value does not get change accidently), in that case we can create local copy of that variable using Module.
- **Whenever we define a function that consists of more than one line then it is desirable to define the contents of function inside a Module.**
- **Example:** Write a function using Module that takes a number say x as input and print the numbers from x to 0 in reverse order with a decrement of 1.

  Mathematica Code:

```
f[x0_ ]:= Module[{x=x0},
          While[x>0, Print[x]; x--]]
f[5]
5
4
3
2
1
0
```

- Note that x is not accessible outside Module, it is local to the Module.
- Let’s print x outside Module

```
  x
x
```

- Notice that x is treated as a symbol outside Module.

APPEND

- Append[*expr*, *elem*] gives *expr* with *elem* appended, i.e., add element at the end of the list.
- For Example: **Append[{a, b, c}, x]**

```
{a, b, c, x}
```

NUMBER FORM

- NumberForm[*expr*, *n*] prints with approximate real numbers in *expr* given to *n*-digit precision.
- For Example: **NumberForm[Pi^9, 10]**

```
29809.09933
```

- NumberForm[*expr*, *n*, *f*] prints with approximate real numbers having *n* digits, with *f* digits to the right of the decimal point.

TABLE FORM

- TableForm[*list*] prints with the elements of *list* arranged in an array of rectangular cells.
- For Example:

```
Table[(i + 45)^j, {i, 3}, {j, 3}]
{{46, 2116, 97336}, {47, 2209, 103823}, {48, 2304, 110592}}
TableForm[%]
Out[2]//TableForm=
    46     2116     97336
    47     2209     103823
    48     2304     110592
```

TABLE HEADINGS

- **For Example:**

```
 TableForm[{{a, b, c}, {ap, bp, cp}},
TableHeadings -> {Automatic, {"first", "middle", "last"}}]
          first      middle      last
1         a          b           c
2         ap         bp          cp
```

- None: no label in any dimension
- Automatic: Successive integer labels in each dimension

NEWTON - RAPHSON METHOD

- Required to find the approximate root of a given function f by iterative method.
- Start with an initial approximation say x0, the kth approximation is given by

$$x_k=x_{k-1}\frac{f(x_{k-1})}{f\prime(x_{k-1})}, \text{ k = 1, 2, …}$$

- No. of iterations as stopping criteria
- Error of Tolerance as stopping criteria
- Maximum no. of iterations and error tolerance as stopping criteria