# *C Programming*

Charudatt Kadolkar

IIT Guwahati

# Trigonometric Tables

We want to print a table of sine function. The output should look like:

```
  0      0.0000
 20      0.3420
 40      0.6428
 60      0.8660
 80      0.9848
100      0.9848
120      0.8660
140      0.6428
160      0.3420
180     -0.0000
```

# Trigonometric Tables

```
main()
{
     int      degrees;
     float    radians;

     degrees   = 0;
     while   ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# return statement

`return` statement *returns* the control to operating system, or in other words, terminates the program

# While Loop

Syntax for `while` loop is

```
while ( condition )
    statement1;
```

or

```
while ( condition )
{
    statement1;
    statement2;
        :
        :
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :  ?
     float    radians;                                    radians   :  ?

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :  0
     float    radians;                                    radians   :  ?

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :  0
     float    radians;                                    radians   :  ?

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees    :   0
     float    radians;                                    radians    :   0.0000

     degrees    = 0;
     while   ( degrees    <=    180   )
     {
          radians    = degrees    * 3.141592654     / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)          );
          degrees    = degrees    + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   0
     float    radians;                                    radians   :   0.0000

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees    :   20
     float    radians;                                    radians    :   0.0000

     degrees    = 0;
     while   ( degrees    <=    180   )
     {
          radians    = degrees    * 3.141592654     / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)         );
          degrees    = degrees    + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
        int     degrees;                                    degrees   :   20
        float   radians;                                    radians   :   0.0000

        degrees   = 0;
        while  ( degrees   <=   180  )
        {
                radians   = degrees   * 3.141592654    / 180.0;
                printf("%3d     %8.4f\n",degrees,sin(radians)       );
                degrees   = degrees   + 20;
        }

        return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   20
     float    radians;                                    radians   :   0.3491

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   20
     float    radians;                                    radians   :   0.3491

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
        int     degrees;                                    degrees   :   40
        float   radians;                                    radians   :   0.3491

        degrees   = 0;
        while  ( degrees   <=   180  )
        {
                radians   = degrees   * 3.141592654    / 180.0;
                printf("%3d     %8.4f\n",degrees,sin(radians)       );
                degrees   = degrees   + 20;
        }

        return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   180
     float    radians;                                    radians   :   2.7925

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
    int     degrees;                                    degrees   :   180
    float   radians;                                    radians   :   2.7925

    degrees   = 0;
    while  ( degrees   <=   180  )
    {
        radians   = degrees   * 3.141592654    / 180.0;
        printf("%3d     %8.4f\n",degrees,sin(radians)       );
        degrees   = degrees   + 20;
    }

    return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   180
     float    radians;                                    radians   :   3.1416

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   180
     float    radians;                                    radians   :   3.1416

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   200
     float    radians;                                    radians   :   3.1416

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Trigonometric Tables

```
main()
{
     int      degrees;                                    degrees   :   200
     float    radians;                                    radians   :   3.1416

     degrees   = 0;
     while  ( degrees   <=   180  )
     {
          radians   = degrees   * 3.141592654    / 180.0;
          printf("%3d      %8.4f\n",degrees,sin(radians)        );
          degrees   = degrees   + 20;
     }

     return;
}
```

# Formatting the Output

```
degrees    = 1234;
printf("%d\n",degrees);                |1234
printf("%3d\n",degrees);               |1234
printf("%4d\n",degrees);               |1234
printf("%5d\n",degrees);               | 1234
printf("%6d\n",degrees);               |  1234
radians    = 123.456789
printf("%f\n",radians);                |123.456789
printf("%4.4f\n",radians);             |123.4568
printf("%6.4f\n",radians);             |123.4568
printf("%8.4f\n",radians);             |123.4567
printf("%10.4f\n",radians);            |  123.456789
```

# Integer Sum

For a positive integer $n$, let

$$s = n + (n-1) + \cdots + 2 + 1$$

We want to write a program which calculates this sum.

# Integer Sum

```
main()  {
     int  sum;
     int  i;
     int  n;

     scanf("%d"    &n);
     sum  = 0;
     i = 1;
     while  ( i <= n )
     {
          sum  = sum  + i;
          i = i + 1;
     }
     printf("Sum    of  first   %d  integers    = %d\n",n,sum);
     return;
}
```

# Factorial Function

For a positive integer $n$,

$$n! = n \times (n-1) \times \cdots \times 2 \times 1$$

We want to write a program which calculates factorial of $n$.

# Factorial Functions

```
main()   {
     int  factorial;
     int  i;
     int  n;

     scanf("%d"    &n);
     factorial    = i = 1;
     while   ( i <= n )
     {
          factorial    = factorial    * i;
          i = i + 1;
     }
     printf("Factorial      of  %d  = %d\n",n,factorial);
     return;
}
```

# Average Heights

A class has $n$ students. We want a program that reads the heights in cms and prints the average.

# Integer Sum

```
main()   {
     int  sum;
     int  i;
     int  n;
     int  height;

     scanf("%d"    &n);
     sum  = 0;
     i = 1;
     while  ( i <= n )
     {
          scanf("%d",    &height);
          sum  = sum  + height;
          i = i + 1;
     }
     printf("Average     Height   = %d\n",n,sum/n);
     return;
}
```

# Exponential Function

For any $x \in \Re$,

$$e^x = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!} + R_n(x)$$

where $R_n(x) = x^{n+1}e^{\xi}/(n+1)!$ for some $0 < c < x$.

If $x$ is sufficiently small, we can approximate exponential function by a polynomial of degree $n$. We can do this in two ways.

1. Add first $n$ terms of the series, for a given $n$.
2. Add first $n$ terms of the series, such that $|R_n(x)| < \epsilon$ for some given $\epsilon > 0$. Also, $|R_n(x)| < ex^{n+1}/(n+1)!$ if $0 < x < 1$.

# Exponential Function

```
float    x,  ex,  t;
int      i,  n;

scanf("%d",    &n);
scanf("%f",    &x);

ex  = 1;
t = 1;
i = 1;
while  ( i <= n )
{
     t = t * x / i;
     ex  = ex  + t;
     i = i + 1;
}

printf("Exp(%f)     = %f\n",   x,  ex);
```

# Exponential Function

```
float    x,  ex,  t,  eps;
int      i;

scanf("%f",    &eps);
scanf("%f",    &x);

ex  = 1;
t = 1;
i = 1;
while  ( 1 > 0 )
{
     t = t * x / i;
     if  ( 2.718282   * t < eps  ) break;
     ex  = ex  + t;
     i = i + 1;
}

printf("Exp(%f)     = %f\n",   x,  ex);
```

# Arrays

Array is a regular arrangement of objects. In C programming,

```
int   marks[10];
```

declaration, creates 10 `int` type variables, puts them in adjacent memories, and calls this *array* of variables by name `marks` . These are numbered from 0 to 9.

We can access the $6^{\text{th}}$ memory location of `marks` array by typing `marks[5]` .

```
marks[5]   = 96;
marks[7]   = 27;
total  = marks[5]   + marks[7];
```

We can enclose any integer expression

```
i = 5;
marks[i]   = 96;
marks[i+2]   = 27;
```

If $\overrightarrow{x} = (x_1, x_2, x_3)$ and $\overrightarrow{y} = (y_1, y_2, y_3)$ are two vectors in 3D, the angle between these is given by

$$\cos^{-1}\left(\frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|}\right)$$

# Angle between vectors

```
float   x[3];
float   y[3];
float   sx,  sy,  sxy;
int     i;

scanf("%f%f%F",     &x[0],   &x[1],   &x[2]);
scanf("%f%f%F",     &y[0],   &y[1],   &y[2]);

sx  = 0;  sy  = 0;  sxy  = 0;
i = 0;
while  ( i < 3 )
{
     sx  = sx  + x[i]*x[i];
     sy  = sy  + y[i]*y[i];
     sxy  = sxy  + x[i]*y[i];
}

angle  = sxy  / sqrt(sx*sy);
```