Lecture 4

math.h
trigonometric

- Defines various mathematical functions
- examples ...

```
#include <math.h>
double acos(double x) Returns the arc cosine of x in radians.
double asin(double x) Returns the arc sine of x in radians.
double atan(double x) Returns the arc tangent of x in radians.
double atan2(doubly y, double x)Returns the arc tangent in radians
    of y/x based on the signs of both values to determine the
    correct quadrant.

double cos(double x) Returns the cosine of a radian angle x.
double cosh(double x) Returns the hyperbolic cosine of x.
double sin(double x) Returns the sine of a radian angle x.
double sinh(double x) Returns the hyperbolic sine of x.
double tanh(double x) Returns the hyperbolic tangent of x.
```

math.h

```
#include <math.h>
double exp(double x) Returns the value of e raised to the xth power.
double log(double x)Returns the natural logarithm (base-e logarithm)
      of x.
double log10(double x)Returns the common logarithm (base-10
     logarithm) of x.
double pow(double x, double y)Returns x raised to the power of y.
double sqrt(double x)Returns the square root of x.
double ceil(double x)Returns the smallest integer value greater than
      or equal to x.
double fabs(double x)Returns the absolute value of x.
double floor(double x)Returns the largest integer value less than or
      equal to x.
```

for e^x use *exp(x)*, never use *pow(exp(1), x)* ...
Examples!

- Branching is deciding what actions to take
- The program chooses to follow one branch or another
- *if()*
- *? :* operator
- *swich()*

if()

If today is Monday I will study C programming

- Based on a concept of *TRUE* or *FALSE*
- *TRUE* is a statement that evaluates to a nonzero value
- *FALSE* evaluates to zero
- Use of relational operators:
  - $>$ greater than - $5 > 4$ is TRUE
  - $<$ less than - $4 < 5$ is TRUE
  - $>=$ greater than or equal - $4 >= 4$ is TRUE
  - $<=$ less than or equal - $3 <= 4$ is TRUE
  - $==$ equal to - $5 == 5$ is TRUE
  - $!=$ not equal to - $5 != 4$ is TRUE
- exapmles ...

Do not use = to test equality, use == !!!

AND and OR

&& and ||

- Used for more complex logical statements
- && - logical AND
- || - logical OR

Basic if syntax

```
if (statement that evaluates to TRUE or FALSE)
  instruction

if (statement that evaluates to TRUE or FALSE)
{
  multi
  line
  instruction
}
```

examples ...

What else?

```
if (statment that evaluates to TRUE or FALSE)
  instruction
else
  another set of instructions

if (statment that evaluates to TRUE or FALSE)
{
  multi
  line
  instruction
}
else
  another set of instructions - could be multiline
```

examples ...

else if()

```
if (statment that evaluates to TRUE or FALSE)
  instruction
else if()
  another set of instructions
else if()
 ...

if (statment that evaluates to TRUE or FALSE)
{
  multi
  line
  instruction
}
else if()
  another set of instructions - could be multiline
```

examples ...

Inline if

?:

- It is like an *if else*
- Might be used within expressions
- The only ternary operator in C

```
if condition is true ? then X return value : otherwise Y value;

int a=5;
int b=1, c=2;
int d = b > c ? a + b : a + c;
```

examples ...

switch()

- Much like nested if else
- Might be more efficient

```
switch( expression )
{
  case expr1:
    instructions;
    break;
  case expr2:
    instructions;
    break;
  default:
    instructions;
}
```

- key word *break*
- key word *default*
- examples...