Lecture 10
CS 137
Fall 2014
by Chantelle Gellert

math.h

```
#include <math.h>
double sin(double x);
double cos(double x); //radians not degrees
double acos(double x); //arc cos

double exp(double x);
double log(double x);
double log10(double x);

double pow(double x, double y);
double sqrt(double x);

M.P.I ~ constant pie
M.E ~ constant e
```

```
a^n + b^n = c^n
a,b,c integers; no solutoin
where n > 2
```

Homer's counter example: $3987^{12} + 4365^{12} = 4472^{12}$

BiSection algorithm

Rouch idea, start with
a, where f(a) > 0
b, where f(b) > 0
compute the midpoint m = (a+b)/2

```
if(f(m) < 0){
  a = m;
}else{
  b = m;
}
```

BiSection Algorithm: always works if f(x) continuous

```
#include <math.h>
#include <assert.h>

double f(double x)
{
        return x-cos(x);
}

double bisection (double a, double b, double epsilon //tell when to stop
        int iterations // max
        )
{
        int i;
        double m, fm;
        assert(f(a)<0.0 && f(b)>0.0);
        assert (epsilon>0);
for (i = 0; i<iterations; i++)
{
        m = (a+b)/2.0;
        fm = f(m);
        if (fabs(fm) < epsilon) //floating points absolute value
                return m;
        if (f(m)>0)
                b = m;
        else a = m;
}

return m;
}
```

```
#include <stdio.h>
double bisection (double, double, double, int);

int main (void)
{
       printf( "%d\n", bisection (-10,10,0.001,10000));
       return 0;
} // => 0.738525 12iterations
```

Fixed point iterations

```
F(x) = x - cos(x) //want x0 such that f(x0) = 0
g(x) = cos(x) //want x0 such that g(x0) = x0
In general, rewrite:
f(x) = 0 to g(x) = x
```

Rough idea:
- make a guess(*)
- compute a new guess: * = g(*)
- repeat until done

```
#include <math.h>
#include <assert.h>

double g(double x)
{
	return cos(x); //only work for this function
}
double fixed (double guess, double epsilion, int iterations)
{
	int i;
	double newguess;
	assert ( eosilon>0.0);
	for (i = 0; i< iterations; i++)
	{
		newguess = g(guess);
		if (fabs ( guess - newguess)<epsilon)
			return newguess;
		guess = newguess;
	}
	return guess;
}


int main (void)
{
	printf ("%g\n", fixed (0.0, 0.001, 10000));
	return 0;
}//0.73876 (17 iterations)
```

Functions as arguments

```
double bisection(double a, double b, double epsilon, int iterations, double
    (*f)(double)){
  //the rest is the same
}

double f0(double x){
  return x - cos(x);
}

double f1(double x){
  x =-x;
  return x*x*x*x-x-1;
}

double bisection(){

}

int main(void){
 printf("%g\n", bisection(10.0,-10.9, 0.00001, 1000,f(f1));

}
```