

Lecture 6.3: Overview

- Functions
 - Math library functions
 - Example `Function.c`
 - Standard library functions
 - Example `Dice.c`

Math Library Functions

- C standard math library
 - standard library supplied with every C compiler
 - predefined mathematical functions
 - e.g. $\cos(x)$, \sqrt{x} , etc.
- Math library header file
 - contains math function declarations
 - `#include <math.h>`
- Math library linker file
 - contains math function definitions (pre-compiled)
 - library file `libm.a`
 - compiler needs to *link* against the math library
 - use option `-llibraryname`
 - Example: `gcc MathProgram.c -o MathProgram -lm`

Math Library Functions

- Functions declared in `math.h` (part 1/2)

- <code>double sqrt(double x);</code>	\sqrt{x}
- <code>double pow(double x, double y);</code>	x^y
- <code>double exp(double x);</code>	e^x
- <code>double log(double x);</code>	$\log(x)$
- <code>double log10(double x);</code>	$\log_{10}(x)$
- <code>double ceil(double x);</code>	$\lceil x \rceil$
- <code>double floor(double x);</code>	$\lfloor x \rfloor$
- <code>double fabs(double x);</code>	$ x $
- <code>double fmod(double x, double y);</code>	$x \bmod y$

Math Library Functions

- Functions declared in `math.h` (part 2/2)

- <code>double cos(double x);</code>	$\cos(x)$
- <code>double sin(double x);</code>	$\sin(x)$
- <code>double tan(double x);</code>	$\tan(x)$
- <code>double acos(double x);</code>	$\arccos(x)$
- <code>double asin(double x);</code>	$\arcsin(x)$
- <code>double atan(double x);</code>	$\arctan(x)$
- <code>double cosh(double x);</code>	$\cosh(x)$
- <code>double sinh(double x);</code>	$\sinh(x)$
- <code>double tanh(double x);</code>	$\tanh(x)$

Math Library Functions

- Program example: `Function.c` (part 1/3)

```

/* Function.c: compute a math function table */
/* */
/* author: Rainer Doemer */
/* */
/* modifications: */
/* 10/28/04 RD initial version */

#include <stdio.h>
#include <math.h>

/* function definition */

double f(double x)
{
    return cos(x);
} /* end of f */

...

```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

5

Math Library Functions

- Program example: `Function.c` (part 2/3)

```

...
/* main function */

int main(void)
{
    /* variable definitions */
    double hi, lo, step;
    double x, y;

    /* input section */
    printf("Please enter the lower bound: ");
    scanf("%lf", &lo);
    printf("Please enter the upper bound: ");
    scanf("%lf", &hi);
    printf("Please enter the step size: ");
    scanf("%lf", &step);

    ...

```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

6

Math Library Functions

- Program example: `Function.c` (part 3/3)

```

...

/* computation and output section */
for(x = lo; x <= hi; x += step)
{
    y = f(x);
    printf("f(%10g) = %10g\n", x, y);
} /* rof */

/* exit */
return 0;
} /* end of main */

/* EOF */

```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

7

Math Library Functions

- Example session: `Function.c`

```

% vi Function.c
% gcc Function.c -o Function -Wall -ansi -lm
% Function
Please enter the lower bound: -0.5
Please enter the upper bound: 1.0
Please enter the step size: .1
f(   -0.5) =  0.877583
f(   -0.4) =  0.921061
f(   -0.3) =  0.955336
f(   -0.2) =  0.980067
f(   -0.1) =  0.995004
f(-2.77556e-17) =  1
f(    0.1) =  0.995004
f(    0.2) =  0.980067
f(    0.3) =  0.955336
f(    0.4) =  0.921061
f(    0.5) =  0.877583
f(    0.6) =  0.825336
f(    0.7) =  0.764842
f(    0.8) =  0.696707
f(    0.9) =  0.62161
f(    1) =  0.540302
%

```

EE

Standard Library Functions

- Standard C library
 - standard library supplied with every C compiler
 - predefined standard functions
 - e.g. `printf()`, `scanf()`, etc.
- C library header files
 - input/output function declarations `#include <stdio.h>`
 - standard function declarations `#include <stdlib.h>`
 - time function declarations `#include <time.h>`
 - etc.
- C library linker file
 - contains standard function definitions (pre-compiled)
 - library file `libc.a`
 - compiler *automatically links* against the standard library (no need to supply extra options)

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

9

Standard Library Functions

- Functions declared in `stdlib.h` (partial list)
 - `int abs(int x);`
 - `long int labs(long int x);`
 - return the absolute value of a (long) integer `x`
 - `int rand(void);`
 - return a random value in the range 0 – `RAND_MAX`
 - `RAND_MAX` is a constant integer (e.g. 32767)
 - `void srand(unsigned int seed);`
 - initialize the random number generator with value `seed`
 - `void exit(int result);`
 - exit the program with return value `result`
 - `void abort(void);`
 - abort the program (with an error result)

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

10

Standard Library Functions

- Random number generation
 - Standard library provides *pseudo* random number generator
 - `int rand(void);`
 - Pseudo random numbers are a sequence of values seemingly random in the range 0 – `RAND_MAX`
 - Computer is a *deterministic* machine
 - Sequence will always be the same
 - Start of sequence is determined by *seed* value
 - `void srand(unsigned int seed);`
 - Trick: Initialize random sequence with current time
 - header file `time.h` declares function `unsigned int time()`
 - `time(0)` returns number of seconds since Jan 1, 1970
 - at beginning of program, use:


```
srand(time(0));
```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

11

Standard Library Functions

- Program example: `Dice.c` (part 1/4)

```

/* Dice.c: roll the dice                                     */
/* author: Rainer Doemer                                   */
/* modifications:                                         */
/* 10/28/04 RD initial version                             */
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

/* function definition */
int roll(void)
{
    int r;

    r = rand() % 6 + 1;
    /* printf("Rolled a %d.\n", r); */
    return r;
} /* end of roll */
...

```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

12

Standard Library Functions

- Program example: `Dice.c` (part 2/4)

```

...
/* main function */

int main(void)
{
    /* variable definitions */
    int i, n;
    int count1 = 0, count2 = 0, count3 = 0,
        count4 = 0, count5 = 0, count6 = 0;

    /* random number generator initialization */
    srand(time(0));

    /* input section */
    printf("Roll the dice: How many times? ");
    scanf("%d", &n);

    ...

```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

13

Standard Library Functions

- Program example: `Dice.c` (part 3/4)

```

... /* computation section */
for(i = 0; i < n; i++)
{ switch(roll())
  { case 1:
    { count1++; break; }
    case 2:
    { count2++; break; }
    case 3:
    { count3++; break; }
    case 4:
    { count4++; break; }
    case 5:
    { count5++; break; }
    case 6:
    { count6++; break; }
    default:
    { printf("INVALID ROLL!");
      exit(10); }
  } /* hctiws */
} /* rof */
...

```

EECS10: Computational Methods in ECE, Lecture 6

(c) 2013 R. Doemer

14

Standard Library Functions

- Program example: `Dice.c` (part 4/4)

```

...

/* output section */
printf("Rolled a 1 %5d times.\n", count1);
printf("Rolled a 2 %5d times.\n", count2);
printf("Rolled a 3 %5d times.\n", count3);
printf("Rolled a 4 %5d times.\n", count4);
printf("Rolled a 5 %5d times.\n", count5);
printf("Rolled a 6 %5d times.\n", count6);

/* exit */
return 0;
} /* end of main */

/* EOF */

```

Standard Library Functions

- Example session: `Dice.c`

```

% vi Dice.c
% gcc Dice.c -o Dice -Wall -ansi
% Dice
Roll the dice: How many times? 6000
Rolled a 1   963 times.
Rolled a 2   995 times.
Rolled a 3  1038 times.
Rolled a 4  1024 times.
Rolled a 5   984 times.
Rolled a 6   996 times.
% Dice
Roll the dice: How many times? 6000
Rolled a 1   977 times.
Rolled a 2  1043 times.
Rolled a 3  1012 times.
Rolled a 4  1001 times.
Rolled a 5   963 times.
Rolled a 6  1004 times.
%

```