EECS 280
Discussion #2

Week of Jan 14

Outline

- **Administrivia**
- The Call Stack or "Where Am I?"
- Recursion
- GDB or "What Went Wrong?"

Announcements

- New Office Hours
- Assignment #2
  - LOTS of thinking, not much coding
  - 2 weeks to complete, but start early
  - Due Jan. 31 @ 11:59 PM

Lessons from PA1

- Submission clarifications
  - Feedback
  - Last submission counts
- Phorums vs. e-mail vs. office hours
- Start early, don't touch a computer till you can do the program by hand

Outline

- Administrivia
- **The Call Stack or "Where Am I?"**
- Recursion
- GDB or "What Went Wrong?"

The Call Stack: A Motivating Example

```
void print( string word )
{
    cout << word << endl;
}

void print_today()
{
    string date = “Jan 16 2008”;
    print( date );
}

void print_hello()
{
    print( “Hi!” );
}
```

```
int main()
{
    print_today();
    print_hello();
    print_hello();
}
```

What is the state of the program?

The Call Stack

- In order to represent *function state*, an activation record contains:
  - local variables/parameters
  - address of the return value
- The call stack uses a list of activation records to represent *program state*

Call Stack Example

- PRINT
  - word = “hi!”

- PRINT_HELLO

- MAIN

- `line 1 of print`
- `line 1 of print_hello`
- `line 2 of hello`

Outline

- Administrivia
- The Call Stack or "Where Am I?"
- **Recursion**
- GDB or "What Went Wrong?"

Recursion

- Style of programming whereby a function calls itself
- Usually requires relatively small amounts code and two special steps:
  - Base case
  - Recursive step

Example: Multiplication

Non-recursive, iterative form:

```
int multiply( int x, int y )
{
   int z = 0;
   while ( y > 0 )
   {
      z += x;
      y -= 1;
   }

   return z;
}
```

Example: Multiplication

Recursive form:

```
int multiply_r( int x, int y )
{
   // base case
   if ( y == 0 )
      return 0;

   // recursive step
   return ( x + multiply_r( x, y - 1 ) );
}
```

multiply_r(5,3);

Call Stack

```
multiply_r(5,3)
   multiply_r(5,2)
      multiply_r(5,1)
         multiply_r(5,0)
         return 0
      return 5 + 0 = 5
   return 5 + 5 = 10
return 5 + 10 = 15
```

Activation Record

```
x=5, y=3

x=5, y=2

x=5, y=1

x=5, y=0

// BASE CASE
```

Exercise: power2

- Write a recursive function "power2" that will compute 2 raised to the power of k (where k is an integer)
- Show the stack for `power2(3)`

One Possible Solution

```
int power2( int k )
{
   // base case
   if ( k == 0 )
      return 1;

   // recursive step
   return ( 2 * power2( k - 1 ) );
}
```

Call Stack: power2(3);

Call Stack

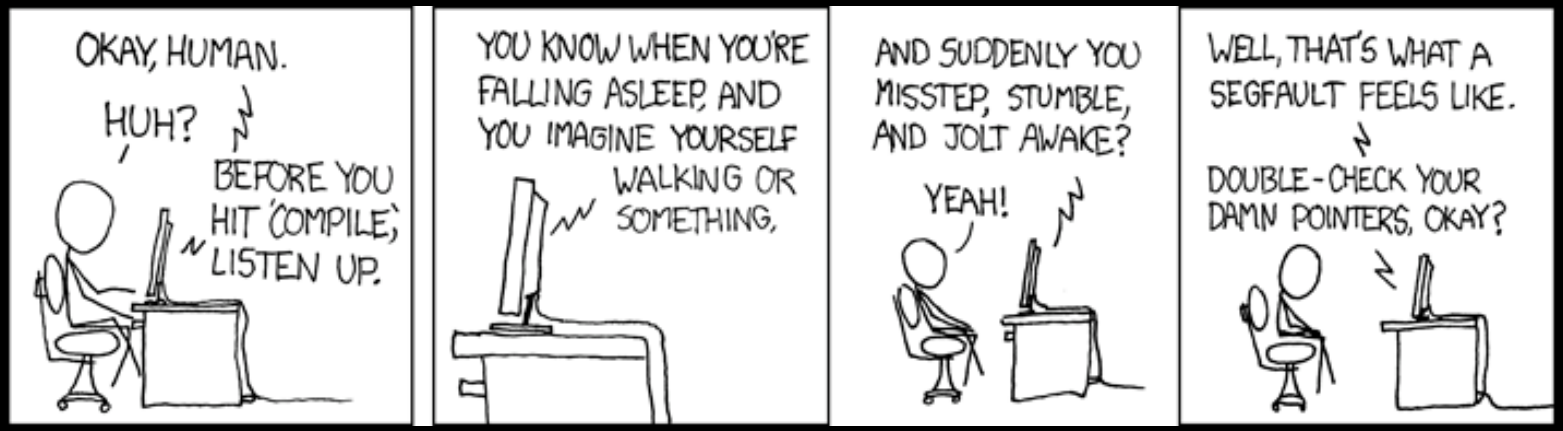
```
power2(3)
    power2(2)
        power2(1)
            power2(0)
            return 1
        return 2 * 1 = 2
    return 2 * 2 = 4
return 2 * 4 = 8
```

Activation Record

```
k=3
k=2
k=1
k=0
// BASE CASE
```

Outline

- Administrivia
- The Call Stack or "Where Am I?"
- Recursion
- **GDB or "What Went Wrong?"**



Motivating Humor

GDB
GNU Project Debugger

- GDB can do four main things
  - Start your program, specifying arguments
  - Make your program stop on certain conditions
  - Examine what happened when your program stopped
  - Change things in your program

Steps to GDB

- Compile with "-g" option
  - g++ -Wall -Werror -m32 px.cpp -o px -g
- Run GDB
  - gdb px
- Issue GDB commands

Useful GDB Commands

- **run**
- **quit**
- **break <function/line>**
- **info break**
- **delete <N>**
- **continue**
- **step**
- **next**
- **up**
- **print <expression>**
- **where**

Questions, Comments, Concerns?

- Have a great MLK :)