CSE 351
Section 7

2/16/12

Agenda

- Midterm Post-Review

Question 1

Write the assembly function for   char *strchr(char *s, char c);
Locates the first occurrence of character c in string s. Returns a pointer to that character, or null pointer if not found. Function should be callable from a module other than the current source file.

```
.text
.globl strchr                  #Make strchr callable elsewhere

strchr:
          pushq %rbp           #Save old frame pointer
          movq %rsp,%rbp       #Set up new frame pointer
loop:
          cmpb (%rdi),%sil     #Check if byte where %rdi points == byte in %sil
          je end
          cmpb $0,(%rdi)       #Check if byte where %rdi points is null character
          je endnull
          addq $1, %rdi        #Increment pointer by 1 to look at next character
          jmp loop
endnull:
          movq $0,%rdi         #Put 0 into %rdi in prep for putting into %rax
end:
          movq %rdi,%rax       #Put resulting pointer in %rax for return
          popq %rbp            #Restore old frame pointer
          ret
```

Question 2

Write the assembly function for the following C code:

```
static int fib(int x){
        if (x <= 1)
                return x;
        return fib(x-1)+fib(x-2);
}
```

```
.text

fib:
           push %rbp                  #Save old frame pointer
           mov %rsp,%rbp              #Set up new frame pointer
           sub $16,%rsp               #Create stack space
           mov %edi,%eax              #Move input argument to eax
           cmp $1,%edi                #Check if input arg is 1 or less
           jle .end                   #If so, end (returning 1 or 0 in eax)
           mov %edi, (%rsp)           #Move argument to top of stack
           sub $1, %edi               #Subtract 1 from argument
           call fib                   #Recurse "fib(x-1)"
           mov %eax, 4(%rsp)          #When we've returned, move returned value to 2nd stack space
           mov (%rsp),%edi            #Move value from top of stack to the input register
           sub $2,%edi                #Subtract 2 from input register
           call fib                   #Recurse "fib(x-2)"
           add 4(%rsp),%eax           #Add fib(x-1) to fib(x-2)
.end:
           add $16, %rsp              #Recover stack space
           pop %rbp                   #Restore old frame pointer
           ret
```

Question 3

- 16-bit signed binary value for 3

  0000 0000 0000 0011

- 16-bit signed binary value for 14

  0000 0000 0000 1110

- 16-bit signed binary value for -14

  1111 1111 1111 0010

- 16-bit signed binary value for 3-14 = -11

  1111 1111 1111 0101

Question 4

a) Assume s is a pointer with the value 0x1000. s points to the string “Hello world!”. What is the address of the letter ‘w’?
   - 0x1006

b) What is a callee saved register?
   - The function being called must save the contents of this register if it wants to use it and restore its value before returning from the function.

c) What is the 32-bit floating point representation for -3.25?
   - $3.25_{10} = 11.01_2$
   - $(-1)^1 * 1.101_2 * 2^1$
   - $S = 1$, $frac = 10100\ldots_2$, $exp = 1+Bias = 1+127 = 128 = 10000000_2$
   - 1 10000000 10100000000000000000000

Question 4

d) (T/F) In 64-bit x86, the first 2 integer arguments are passed in registers, the remainder on the stack.
   - False, the first six are passed in registers.

e) (Big/Little) endian: The number 0xdeadbeef is stored in memory as byte 0: 0xef, byte 1: 0xbe, byte 2: 0xad, byte 3: 0xde
   - Little Endian

f) (T/F) The return value from this function is always 1.
   int foo() { int x = random(); int y = random(); unsigned ux = x; unsigned uy = y; return ux + uy == x + y;}
   - True. x and y get cast as unsigned in the == comparison since ux and uy are unsigned.

g) The smallest signed 16-bit integer is?
   - $-32768_{10} = 1000\ 0000\ 0000\ 0000_2$ = 0x8000