EC312 SPRING 2015 6-week Practice Exam

Please note the following:
1) This is the 6-week exam that was given for EC312 in Spring 2014.
2) The course has evolved since Spring 2014, and there may be topics that we have covered in more detail this semester than you’ll see tested here in the practice exam.
3) There is NO guarantee that the 6-week exam for this semester will resemble this practice exam. Working through this practice exam should not be deemed as sufficient preparation for the upcoming 6-week exam.
6 WEEK EXAM

NAME:

ALPHA: _________________________________

SECTION: _____________________________

1. This is individual work.
2. SHOW ALL WORK!
3. Write legibly to receive credit.
4. Turn in your equation sheet.

---

**ASCII Table for Printable Characters**

<table>
<thead>
<tr>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
<th>Dec Hex Char</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 20</td>
<td>40 30</td>
<td>48 60</td>
<td>64 A0</td>
<td>80 C0</td>
<td>96 E0</td>
<td>112 70</td>
<td>128 79</td>
<td>144 80</td>
<td>160 89</td>
<td>176 98</td>
</tr>
<tr>
<td>33 21</td>
<td>41 31</td>
<td>49 61</td>
<td>67 B0</td>
<td>83 D0</td>
<td>99 F0</td>
<td>115 80</td>
<td>131 99</td>
<td>147 00</td>
<td>163 09</td>
<td>179 08</td>
</tr>
<tr>
<td>34 22</td>
<td>42 32</td>
<td>4A 62</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>35 23</td>
<td>43 33</td>
<td>49 63</td>
<td>68 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>36 24</td>
<td>44 34</td>
<td>4A 64</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>37 25</td>
<td>45 35</td>
<td>4B 65</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>38 26</td>
<td>46 36</td>
<td>4C 66</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>39 27</td>
<td>47 37</td>
<td>4D 67</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>40 28</td>
<td>48 38</td>
<td>4E 68</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
<tr>
<td>41 29</td>
<td>49 39</td>
<td>4F 69</td>
<td>66 70</td>
<td>82 70</td>
<td>89 70</td>
<td>110 69</td>
<td>127 78</td>
<td>143 69</td>
<td>160 58</td>
<td>176 47</td>
</tr>
</tbody>
</table>

---

**4-bit pattern**

<table>
<thead>
<tr>
<th>hex digit</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0010</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0011</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0101</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0110</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0111</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1010</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1011</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1101</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1110</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1111</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Score</th>
<th>Question</th>
<th>Points</th>
<th>Score</th>
<th>Question</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>17</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>10</td>
<td>2</td>
<td>2</td>
<td>18</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>11</td>
<td>4</td>
<td>4</td>
<td>19</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>12</td>
<td>4</td>
<td>4</td>
<td>20</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>13</td>
<td>4</td>
<td>4</td>
<td>21</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>14</td>
<td>2</td>
<td>4</td>
<td>22</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>13</td>
<td>15</td>
<td>6</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>16</td>
<td>4</td>
<td>Total</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Lesson 1 – Computer System Review

1. (2pts) The address 0x080483ac is written in _________ notation and represents ___________ bits.
   a. Binary, 8
   b. Decimal, 8
   c. Decimal, 32
   d. Hexadecimal, 8
   e. Hexadecimal, 32

2. (2pts) Looking for characters, you type the command x/xb 0x08048a9b and it returns 0x56. If the memory located at the address 0x08048a9b is in fact a character, what character does it represent?

3. (2pts) Registers are located in a computer’s
   a. main memory
   b. CPU
   c. hard disk drive.
   d. power supply

4. (2pts) Draw the schematic symbol for a transistor and a diode.

   Transistor

   Diode

Lesson 2: C Programming; Digital Logic

5. (3 pts) Fill in the table below with the amount of memory each data type requires. Express your answers in terms of both bytes and bits.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Number of Bytes</th>
<th>Number of Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>An integer variable (int)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A floating point variable (float)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A character variable (char)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6. (6pts) Complete the truth table for the given logic circuit. X and Y represent two intermediate values and Z represents the output of the circuit.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7. (13pts) Match the term on the left with its appropriate description on the right:

- __ =  a. the format string for an integer value
- _ high level language  b. the C assignment operator
- _ scanf  c. program which allows the debug of machine code
- _ gcc  d. translates assembly language into machine language
- _ machine language  e. allows the use of functions from various header files
- _ nano  f. allows the program to receive keyboard input
- _ assembler  g. the C escape sequence for a new line
- _ #include <..>  h. the C programming language
- _ assembly language  i. program that converts source code to object code
- _ `%d`  j. a UNIX text editor, can be used to write C programs
- _ printf  k. function which displays text to the monitor
- _ `\n`  l. instructions expressed as bits, also called object code
- _ gdb  m. English-like words that represent machine code
Lesson 3 – Registers and Memory

8. (2pts) **eip** is a __________ that points to the __________.
   a. location in RAM, top of the stack
   b. location in RAM, bottom of the stack
   c. location in RAM, next instruction to be executed
   d. register, top of the stack
   e. register, bottom of the stack
   f. register, next instruction to be executed

9. (2pts) **esp** is a __________ that points to the __________.
   a. location in RAM, top of the stack
   b. location in RAM, bottom of the stack
   c. location in RAM, next instruction to be executed
   d. register, top of the stack
   e. register, bottom of the stack
   f. register, next instruction to be executed

10. (2pts) **ebp** is a __________ that points to the __________.
    a. location in RAM, top of the stack
    b. location in RAM, bottom of the stack
    c. location in RAM, next instruction to be executed
    d. register, top of the stack
    e. register, bottom of the stack
    f. register, next instruction to be executed

11. (4 pts)
    a. (2 pts) What type of language is depicted in the screenshot above?
    b. (2 pts) Describe what the line of code at address 0x080483c4 accomplishes.
12. (4 pts) Recall that in RAM you have stored the object code for your program as well as additional memory allocated for your variables within the program.

You type into the debugger the command `i r ebp` and get the result `0xbffff7e8`. Upon further review of the assembly code you determine that a string is stored in memory `[ebp-12]`. What is the string?

String: ______________________________

---

Lesson 4 - Arrays

13. (4 pts) Use the array declaration to answer the questions.

```c
float voltages[6]={ 0.4, 0.75, 1.23, 2.1, 3.1, 4.2};
```

a. (2 pts) How many bytes are allocated for this array?

b. (2 pts) What value is stored in `voltages[5]`?

---

Lesson 5 – Strings and Pointers

14. (2pts) Given the following declarations, what would be the C statement to assign `ptr_Data` the address of integer `Data`? (Circle the correct answer)

```c
int  Data;
int  *ptr_Data;
```

a) `*ptr_Data = &Data;`
b) `&ptr_Data = &Data;`
c) `ptr_Data = &Data;`
d) `ptr_Data = Data;`
e) `&ptr_Data = *Data;`
15. (6pts) Given the following C statements and memory map below, what would be the result of the printf statement?

```c
int a = 11;
int b[2] = {20,12};

printf("The address of array b is %x \n", b);
```

Result:

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xBFFFF8F0</td>
<td>a0</td>
<td>0xBFFFF8FA</td>
<td>00</td>
</tr>
<tr>
<td>0xBFFFF8F1</td>
<td>84</td>
<td>0xBFFFF8FB</td>
<td>00</td>
</tr>
<tr>
<td>0xBFFFF8F2</td>
<td>04</td>
<td>0xBFFFF8FC</td>
<td>0c</td>
</tr>
<tr>
<td>0xBFFFF8F3</td>
<td>08</td>
<td>0xBFFFF8FD</td>
<td>00</td>
</tr>
<tr>
<td>0xBFFFF8F4</td>
<td>f8</td>
<td>0xBFFFF8FE</td>
<td>00</td>
</tr>
<tr>
<td>0xBFFFF8F5</td>
<td>f8</td>
<td>0xBFFFF8FF</td>
<td>00</td>
</tr>
<tr>
<td>0xBFFFF8F6</td>
<td>ff</td>
<td>0xBFFFF900</td>
<td>e0</td>
</tr>
<tr>
<td>0xBFFFF8F7</td>
<td>bf</td>
<td>0xBFFFF901</td>
<td>0c</td>
</tr>
<tr>
<td>0xBFFFF8F8</td>
<td>14</td>
<td>0xBFFFF902</td>
<td>00</td>
</tr>
<tr>
<td>0xBFFFF8F9</td>
<td>00</td>
<td>0xBFFFF903</td>
<td>b8</td>
</tr>
</tbody>
</table>

16. (4pts) Will the following C source code compile without error? If not, explain why. If so, describe what output the program will produce.

```c
#include<stdio.h>
void sum( int x, int y ,int z)
{
    int total;
    total = x + y + z;
    printf("\nThe total of the numbers is: %d\n\n",total);
}
int main()
{
    int num1=7, num2=1, num3=9;
    sum( num2 ,num3 );
}
```
Lesson 7 – Buffer Overflow Introduction

17. (2pts) What feature of the C language makes a buffer overflow attack possible?
   a. C has a garbage collector
   b. C performs automatic bounds checking
   c. C does not perform automatic bounds checking
   d. C allows you to access registers

18. (4pts) When the `printout` function is called in `main` from the following code sample, the stack pictured below is created.

```c
#include<stdio.h>
void printout()
{
    int j;
    char input_string[10];

    printf("Input a string: ");
    scanf("%s", input_string);
    for(j=0; j < 10; j=j+1)
    {
        printf("%s\n", input_string);
    }
}

int main()
{
    printout()
}
```

Assume there is no padding (i.e. extra bytes in between variables) when the frame is created. What is the largest number of characters the user can enter without overwriting any portion of the return address? Show all assumptions and work.
19. (2pts) Suppose a user types `./a.out "Go Navy" 6 "cyber2"` at the command line. What will be the value of `argc`?
   a. 3  
   b. 4  
   c. 5  
   d. None of the above.

Lesson 8 – File I/O and Permissions

20. (2pt) In which direction does the heap grow?
   a. From the bottom (larger memory address) up (to a smaller memory address).
   b. From the top (smaller memory address) down (to a larger memory address).
   c. It depends on the corresponding number and types of variables currently allocated on the stack.
   d. It depends on the prolonged effects of solar and liquefied additives combined with the chemical makeup of the heap.

21. (2pt) Suppose we run a program that allocates 10 bytes of ram and assigns it to a pointer using the command shown
    ```c
    ptr=(char *) malloc ( 10 );
    ```
    What is a possible value for `ptr` after executing the command given the value of the instruction pointer and stack pointer are:
    ```c
eip=0x0804848a
esp=0xbffff7c0
```
   a. 0x08048000  
   b. 0x0804fff0  
   c. 0xbffff7d0  
   d. none of the above

Lesson 9 – Privilege Management

22. (4pts) Looking at the long listing of the files below.

```
-rw-r--r-- 1 root root 0 2014-02-07 10:27 northside
-rw------- 1 instructor midshipman 166 2014-02-05 17:26 notes
drwx------ 2 midshipman midshipman 4096 2014-02-05 17:27 orbit-midshipman
-rw-r-xr-x 1 root midshipman 95 2014-02-07 10:28 providence.exe
```
   a. (2pts) Who is the owner of the `notes` file?
   b. (2pts) What permissions do other users have for the `notes` file?