Name:_____

Problem 1: (20 points) Short Answers

a) **(3 points)** Which of the following is not a job of the linker? (Choose one answer)

_____relocation

_____compute branch offsets

_____combine .o files

_____resolve external symbols

b) **(5 points)** Consider the following assembly language program segment, which loads $t0 with the larger of $a1 and an integer labeled by value.

```
.data
value: .space 4    #4 bytes storage for variable value

.text
lui $at, upper half of value
lw $t1, lower half of value ($at)
slt $at, $t1, $a1
beq $at, $0, t1greater
add $t0, $0, $a1
j gotmax
t1greater:
add $t0, $0, $t1
gotmax:
...
```

The table below lists some of the statements in the program segment. Indicate which of the statements listed below will be represented by an entry in the relocation table.

Statement	Will it contribute an entry to the relocation table?
`lui $at, upper half of value`	
`lw $t1, lower half of value($at)`	
`beq $at,$0,t1greater`	
`j gotmax`	
`add $t0, $0, $t1`	

c) (3 points) Moore's Law and Registers

Name:_____

MIPS was invented in 1985 with 32 integer registers. According to Moore's Law, named after Intel founder Gordon Moore, the number of transistors per microprocessor doubles every 1.5 years. Thus, microprocessors should have approx 32,000 times the number of transistors today as they could in 1985. It would seem that we could easily build microprocessors with, say, 1K registers or more.

Select **all the reasons** why MIPS has **not** increased the number of integer registers from 32 to 1K.

1. There is no need for more than 32 registers, as compilers have difficulty using the 32 registers in the MIPS architecture now.
2. Due to the importance of binary compatibility, new MIPS processors must be able to execute old MIPS instructions, and it would be very hard to find room for 1K registers in the original MIPS instruction format.
3. 32 registers are much smaller than 1K registers, and since smaller is faster, the 32 registers makes it easier to build fast microprocessors than if there were 1K registers.
4. Moore's Law applies to Intel microprocessors, not MIPS microprocessors hence the hypothesis is false. The MIPS chip would be too expensive if it had 1K registers.

d) **(3 points)** What do the acronyms CISC and RISC stand for? What is the difference between the two?

e) **(3 points)** Name the 4 regions of a program's memory space

f) **(3 points)** Name the 5 classic components of a computer

Name:_____

Problem 2: (15 points) Representations

a) **(5 points)** Translate the decimal number 33.0625 to a 32 bit IEEE 754 single precision floating-point number

b) **(5 points)** Translate the 32 bit IEEE 754 single precision floating-point number `0xBE500000` to decimal fraction number. 2 points partial credit if the number is correct but not in decimal fraction form.

c) **(5 points)** Translate the hexadecimal number `0x3C041001` to a MIPS instruction using register names, e.g. $s0, $t0, …

Name:_____

Problem 3: (45 points) Compilation

a) (20 points) Frabjous Day

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe:
All mimsy were the borogoves,
And the mome raths outgrabe.

Beware the Jabberwock, my son!
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!

He took his vorpal sword in hand:
Long time the manxome foe he sought –
So rested he by the Tumtum tree,
And stood awhile in thought.

And, as in uffish thought he stood,
The Jabberwock, with eyes of flame,
Came whiffing through the tulgey wood
And burbled as it came!

001, 010! 001, 010! And through and through
The vorpal blade went snicker-snack!
He left it dead, and with its head
He went galumphing back.

"And hast thou slain the Jabberwock?
Come to my arms, my beamish boy!
O frabjous day! Callooh! Callay!"
He chortled in his joy.

Twas brillig, and the slithy toves
Did gyre and gimble in the wabe:
All mimsy were the borogoves,
And the mome raths outgrabe.

As one might imagine, the hero's father was rather nervous while his son was out hunting. Alice helpfully offers to write a program to help him determine if the day will be frabjous or most unhappy. If his beamish boy remembers to carry the vorpal sword and first happens across the jabberwock, the day will be quite frabjous. On the other hand, if the boy should happen to stumble across a frumious Bandersnatch, he is doomed.

Alice wrote the following program in C and converted the `snickersnack` function to MIPS assembly language obeying standard MIPS calling conventions. Unfortunately, she was called away by the Queen of Hearts before she could complete the `isFrabjousDay` function. Help the father by finishing the code on the next page. Be sure to save and restore only the necessary registers so the code executes quickly.

```
int snickersnack(int target, int weapon)
{
     return target && weapon;
}
```

Name:_____

```
int isFrabjousDay(int bandersnatch, int jabberwock, int
vorpal)
{
    int frabjous = 0;
    if (snickersnack(jabberwock, vorpal) != 0) frabjous = 1;
    else if (bandersnatch != 0) frabjous = -1;
    return frabjous;
}
```

Write your MIPS assembly language code here for *`isFrabjousDay`*:

Name:_____

b) **(25 points) Minding your p's and q's**

Below is a segment of C code:

```
char *p, *q, x = 'a', y = 'b';
p = &x;
q = &y;
*q = *p++;
q = p;
```

Translate the above C code into MIPS assembly code. Assume that the variable are all stored on the stack in the order that they are declared in the code (i.e. p is stored at 0($sp) and so on). The integer ASCII code for ‘a’ and ‘b’ are 97 and 98, respectively. For those that are interested, the ASCII code for ‘A’ and ‘B’ are 65 and 66, respectively.

Name:_____

Problem 4: (40 points) MIPS Reverse Compilation

Consider the following MIPS assembly code:

```
stewie:
      lb $t2, 0($a0)
      beq $t2, $0, quagmire
      slti $t0, $t2, 97
      bne $t0, $zero, brian
      addi $t1, $zero, 122
      slt $t0, $t1, $t2
      bne $t0, $zero, brian
      addi $t2, $t2, -32
      sb $t2, 0($a0)
brian:
      addi $a0,$a0, 1
      j stewie
quagmire:
      jr $ra
```

a) **(20 points)** Assume that an array A is located somewhere in memory, and the base address of A is stored in `$a0`. Describe concisely what the code does. Specifically, at the end of execution, what will be stored in array A? You will be given partial credit if you simply translate the above code into C code, without describing exactly what it does.

Name:_____

b) **(20 points)** Convert the following instructions from the code above into 32 bit hexadecimal number. Assume that the address of the first instruction (`lb $t2, 0($a0))` is located at address `0x00400028`.

(5 points) `beq $t2, $0, quagmire`

(5 points) `j stewie`

(5 points) `lb $t2, 0($a0)`

(5 points) `addi $t1, $zero, 122`

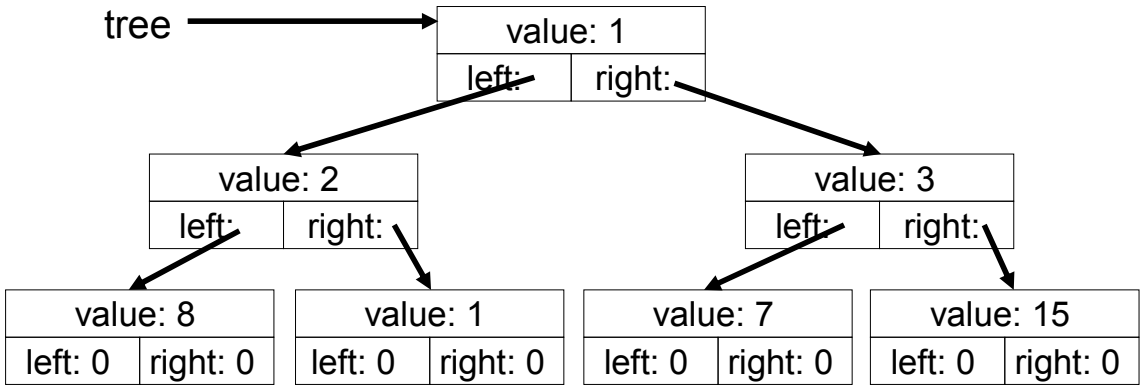
Name:_____

Problem 5: (45 points) Binary Tree Data Structure

Assume that we are using a binary tree as a data structure for some random task. The binary tree holds an integer value and has two pointers; one to the left subtree and one to the right subtree. The TreeNode structure for this data structure is as follows:

```
struct TreeNode {
     int value;
     struct TreeNode * left;
     struct TreeNode * right;
     } * BinaryTree;
```

An example binary tree is as follows:



In the above figure, we have a three level, balanced binary tree. The variable "`tree`" is a pointer to the root `TreeNode`. It is not necessary for the tree to be balanced. For instance, we could remove the bottom right node with the value 15 and make this an unbalanced tree. Each `TreeNode` can have 0, 1 or 2 children, but no more than 2.

Each `TreeNode` structure holds some value that is somehow pertinent to the random problem, but not important for you to solve this problem. The `left` and/or `right` variable holds the value of `NULL`, which is equal to 0, if it does not have any subtrees.

a) **(5 points)** Assuming we compile to the MIPS processor discussed in class, what is the size of a single TreeNode structure, i.e. what value would a call the `sizeof(struct TreeNode)` return?

Name:_____

b) **(15 points)** Given the following data layout in memory and knowing that the variable `tree = 0x1000`, draw a sketch similar to the one above of the binary tree represented below. All the values are noted in hexadecimal format.

Address	Value
0xFFFF	. . .
	0x3AD0
	0xB844
0xEAB0	0x0A00
	. . .
	0x0000
	0x0000
0xB844	0xEAB0
	. . .
	0x0000
	0x0000
0x9400	0xEAB0
	. . .
	0x9400
	0x0000
0x64AC	0x1000
	. . .
	0x0000
	0x0000
0x3AD0	0x64AC
	. . .
	0x64AC
	0xEAB0
0x1000	0x9400
	. . .
	0xB844
	0xEAB0
0x0030	0x1000

Name:_____

c) **(25 points)** Below is a recursive function called `sum_tree` that traverses the tree and returns the sum of all of the values in the tree.

```
int sum_tree(BinaryTree tree)
{
        int sum = tree->value;
        if(tree->left != NULL)
             sum += sum_tree(tree->left);
        if(tree->right != NULL)
             sum += sum_tree(tree->right);
        return sum;
}
```

Convert `sum_tree` to MIPS assembly. You must exactly translate the code above, i.e. you should not try to optimize it and it must be recursive. Also, you must follow all of the MIPS procedure conventions. Failure to do either of these will result in a significant loss of points.

Name:_____

Extra Credit: (5 bonus points) History of the Computing World: Part I

1. The following people have all won Turing Awards. Describe the primary reason that they won the award, i.e. briefly describe what is on their Turing award citation.

a) John Backus

b) Stephen A. Cook

c) John Cocke

d) Dennis M. Ritchie

e) Donald E. Knuth

2. Church hypothesized that "A function of positive integers is effectively calculable only if recursive.". While Turing hypothesized "Logical computing machines [Turing's expression for Turing machines] can do anything that could be described as "rule of thumb" or "purely mechanical". Turing later proved that these theses were equivalent. For an A+ in this class and a PhD degree in CS (after fulfilling all the other class requirements), prove the Church-Turing thesis. Feel free to use extra sheets of paper.