MIPS has “R-type” and “I-type” instructions that are 32 bits. Assume $rX$ means register X. Show the hex machine code equivalent for the following instructions. The op code for R-type is always zero, and the function code for addition is 32 (decimal). The op code for add immediate is 8 (decimal).

### R-Type

<table>
<thead>
<tr>
<th>op</th>
<th>rs</th>
<th>rt</th>
<th>rd</th>
<th>sham</th>
<th>funct</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 bits</td>
<td>5 bits</td>
<td>5 bits</td>
<td>5 bits</td>
<td>5 bits</td>
<td>6 bits</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>I-Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>op</td>
</tr>
<tr>
<td>6 bits</td>
</tr>
</tbody>
</table>

1) 3 points. Convert to hex.

- add $r17, r3, r3

2) 3 points. Convert to hex

- addi $r6, $r6, -15

3) 5 points. Convert the following hex value into MIPS assembly

- 0x2084fff4

4) 5 points. Convert the following C code into the MIPS assembly language equivalent

```c
a0 = a0 + 9;
if (a0 != a1)
    a3 = a2 + s1;
```

5) 3 points. On a MIPS machine, how many bytes would the following structure be?

```c
struct
{
    int x;
    char y;
    int z;
};
```

6) 4 points. Assume $a0$ is a pointer to one of the structures in the previous question. Write MIPS code to do this:

```c
a0->z = 10;
```
For the following questions, assume that the **data segment starts at 0x10010000**, and any **code starts at 0x40000000**. The MIPS assembly language sometimes converts a single machine instruction into two simpler instructions; go ahead and assume that this does not happen, and each instruction is only 32 bits.

<table>
<thead>
<tr>
<th>.data</th>
<th>7) 5 points. What is in $ra, after the jal?</th>
</tr>
</thead>
<tbody>
<tr>
<td>a:</td>
<td>.word 4</td>
</tr>
<tr>
<td>b:</td>
<td>.word 5</td>
</tr>
<tr>
<td>c:</td>
<td>.word 8</td>
</tr>
<tr>
<td>d:</td>
<td>.word 51</td>
</tr>
<tr>
<td></td>
<td>.word 0</td>
</tr>
<tr>
<td>a:</td>
<td>.word 4</td>
</tr>
<tr>
<td>b:</td>
<td>.word 5</td>
</tr>
<tr>
<td>c:</td>
<td>.word 8</td>
</tr>
<tr>
<td>d:</td>
<td>.word 51</td>
</tr>
<tr>
<td></td>
<td>.word 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8) 3 points. What is in a1, after this instruction?</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw $a1, d</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9) 3 points. What is in a1, after this instruction?</th>
</tr>
</thead>
<tbody>
<tr>
<td>la $a1, d</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7) 5 points. What is in $ra, after the jal?</th>
</tr>
</thead>
<tbody>
<tr>
<td>addi $a0, $0, 55</td>
</tr>
<tr>
<td>addi $a1, $a0, 23</td>
</tr>
<tr>
<td>beq $a3, $a4, skip</td>
</tr>
<tr>
<td>jal cosine</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>10) 10 points. In C, the strlen function returns the length of a string. For example, strlen(&quot;example&quot;) returns the number 7. Show how you would write strlen as a MIPS subroutine.</th>
</tr>
</thead>
<tbody>
<tr>
<td>11) 10 points. Write a subroutine addArrays(int length, int *a, int *b, int *c) which adds each element of a and b together to create c. In other words, c[i] = a[i] + b[i] for each value i. The first argument is the length of all the arrays.</td>
</tr>
</tbody>
</table>
12) 10 points. Write a MIPS assembly subroutine for this C code. Make sure not to forget to do something important with a register. The “coffee” function is defined elsewhere. You only need to do the code for “pizza.”

```c
int pizza(int x, int y)
{
    x = x + 2;
    y = y 4;
    return coffee(x, y);
}
```

13) 5 points. Write MIPS assembly to call your subroutine, with the C code.

```mips
s0 = pizza(5, 6);
```

14) 3 points. Suppose you have a pipelined machine. Each cycle is 210ps, and there are five stages in the pipeline. How long does it take to complete one instruction?

15) 5 points. If there are no control hazards or data hazards, on average your pipelined machine completes one instruction every X ps. What is X?

16) 5 points. The following MIPS code has a data hazard. Indicate what causes the hazard, and modify the code so that it will run faster.

```mips
add $s3, $t1, $a0
addi $s5, $t1, $t2
add $s4, $s5, $t2
```
17) 3 points. In the following C code, what is printed for A?

```c
int a = 5;
int b = 10;
void exam2a(int *x, int *y)
{
    int temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

int main()
{
    exam2a(&a, &b);
    printf("A is %d\n", a);
}
```

18) 3 points. In the following C code, what is printed for A?

```c
int a = 5;
int b = 10;
void exam2b(int x, int y)
{
    int temp;
    temp = x;
    x = y;
    y = temp;
}

int main()
{
    exam2b(a, b);
    printf("A is %d\n", a);
}
```

19) 10 points. Write MIPS assembly code for the exam2a subroutine above.

20) Secret bonus points question. How many secret bonus points is the secret bonus point question worth?