CMPU-224 Lab5 Quiz Solutions

Spring 2025

Name: Solution

This is a closed book, closed notes quiz. No electronic devices are allowed. You have until 3:30pm to complete the quiz. There are a total of 3 questions and 10 points.

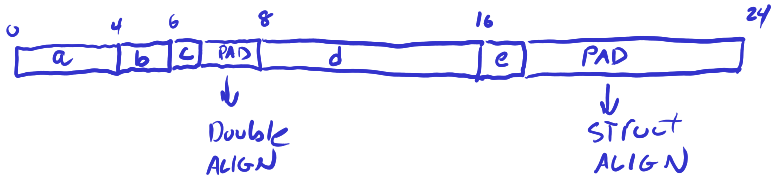
There should be enough space on the quiz for your answers. If you need more space to work out a problem, blank paper will be available, just ask.

If you finish with time remaining, raise you hand and I will come and collect your quiz. You may then work on the lab assignment.

Good Luck!

1. Consider the following structure declaration:

```
struct question {
    char  *a;
    short  b;
    char   c;
    double d;
    char   e;
};
```



   (a) (1 point) Give the total size of the structure in bytes (i.e., the value of the `sizeof(struct question)` expression).

   `sizeof(struct question)` returns **24**

   (b) (2 points) Assume you have the following C code utilizing the above structure:

```
long prob(struct question *x) {
    return x->c;
}
```

   Fill in the missing blanks below for the assembly language implementation of the above function.

```
prob:
    lb a0, 6(a0)
    ret
```

   (c) (1 point) If the structure declaration above was rearranged to minimize the space of the structure, what would be the new size of the structure?

   Total size of rearranged and minimized `struct` in bytes: **16**

2. (2 points) You have the following two-dimensional array declared as follows:

   `int matrix[12][4];`

   Assume the start of the array is at address `0x5000`.

   You have an array access `matrix[x][y]`. What are the indices (`x` and `y` below) of the array that refers to the array element at address `0x50AC`?

   `x is` **10**

   `y is` **3**

3. (4 points) Consider the following C function that accesses an array of 32-bit integers and checks a condition:

```
// Function to swap two integers using pointers
void swap(int *a, int *b) {
    int temp;    // Holds the temporary value
    temp = *a;   // Put the value at address 'a' into temp
    *a = *b;     // Put the value at address 'b' into address 'a'
    *b = temp;   // Put the value in temp into address 'b'
}
```

   Below is the corresponding assembly code. Fill in the missing assembly instructions to make the code function correctly.

```
swap:

  lw a5, 0(a0)

  lw a4, 0(a1)

  sw a4, 0(a0)

  sw a5, 0(a1)
```

Solution:

- **Blank 1:** `slli t0, a1, 2` (Multiplies index by 4 to get the byte offset).
- **Blank 2:** `lw t1, 0(t0)` (Loads the 32-bit integer from memory).
- **Blank 3:** `blt t1, t2, .END` (Branches to the false condition if `arr[index] < 50`).