

## Quiz 2 Solution

### CSE 112 Computer Organization

---

#### Question1:

```
cin >> n;
```

```
for(int i = n ; i != 0; i--)
{
    for(int j = n - i + 1; j != 0; j--)
    {
        cout<<" * "<<" " <<;
    }
    cout << endl;
}
```

```
-----
in r1
mov r2 $1                //Decrement for both loops
mov r3 r1                // Iterator for loop_1

loop_1:
    beq r3 $0 loop_1_exit //exit condition for loop_1
    mov r4 r1             //iterator for loop2 (r4 = n)
    add r4 r4 r2          //iterator for loop2 (r4 = n+1)
    sub r4 r4 r3          //iterator for loop2 (r4 = n+1-i)
    loop_2:
        beq r4 $0 loop_2_exit //exit condition for loop_2
        out "*"             // print new line
        sub r4 r4 r2       // loop_2 iterator update
        beq r2 $1 loop_2   // Unconditional Jump
    loop_2_exit:

    out "\n"              // Print new line
    sub r3 r3 r2          // loop_1's iterator update
    beq r2 #1 loop1       // unconditional jump
loop_1_exit:
```

**[20 marks total. Give partial marks in proportion to how much of the code is correct.]**

**Partial Marking criteria:**

Please go through every student's solution individually, and give marks, out of 20, in proportion to the correct amount of code. If the student's code is not correct, then award marks as follows:

- If pseudocode exists: +1.5
- If the pseudocode is correct: +1.5
- If the pseudocode matches assembly: +1.5
- If the code is reading the input from the user: +0.5
- If the code is printing \* : +0.5
- If the code is printing '\n': +0.5
- If the first loop exists: +1.5
- If the second loop exists: +1.5
- If the loops are nested: +1
- If the first loop performs correct number of iterations (= number provided as input): +1.5
- If the first loop prints "\n" in each iteration: +1.5
- If the second loop performs correct number of iterations (= the iterator of first loop): +1.5
- If the second loop prints "\*" in each iteration: +1.5

**Question 2:**

```
int my_add(int x, int y)
{
    int temp = x + y;
    return temp;
}
```

```
int my_sub(int x, int y)
{
    int temp = x - y;
    return temp;
}
```

```
int foo ()
{
    int a;
    int b;
    int c;

    a = 10;
    b = 20;
    c = my_add(a, b);

    return 2*c;
}
```

```
int main()
{
    return foo();
}
```

```

1  my_add:  push (a)_r1__      // save a caller saved register
2           add r2 r3 r4
3           pop r1
4           (b)_mov__ r15 r1    // return statement

5  foo:     mov r3 (c)_#10__    // move an immediate value
6           mov r4 #20
7           push (d)_r1__      // save a caller saved register
8           (e)_br1__ my_add    // call my_add
9           pop r1
10          add r2 (f)_r2__ r2   // store the result
11          mov r15 (g)_r1__     // return statement

12 main:    push (h)_r1__      // save return address of main func
13          brl foo
14          (i)_pop__ r1        // restore return address of mainfunc
15          (j)_mov r15 r1__     // return statement

```

**[20 x 1 = 20 marks. No partial marking. 1 for each correct answer]**

**Each blank is of 2 marks. If the contents are correct award 2 marks else 0.**

---