CS311 - Computer Architecture - Spring 2017
Project 1 – Three MIPS programs
50 points
Due: Feb 15.

For your first programming project, you are to write MIPS programs to solve the following three problems:

1. One way to generate a so-called pseudo-random number sequence $X_1, X_2, X_3, \ldots$ is to use the following recurrence relation:

$$X_{i+1} = (2017X_i + 311) \bmod 2^{20}.$$

Write a program which prompts the user for an initial *seed* value X_0 and a number n and output n pseudo-random numbers using the equation above. You should output the n values one per line.

2. The following program calculates (very crudely) the square root of an integer to the 10^{-6} place:

```
void main()
{
    int x, d, div;

    div = 0;
    cin >> x;
    d = getDigit(x, div);
    cout << d << ".";
    for(int i=0; i<6; i++) {
        x -= (div+d)*d;
        x *= 100;
        div += 2*d;
        div *= 10;
        d = getDigit(x, div);
        cout << d;
    }
    cout << endl;
}
```

```
int getDigit(int x, int y)
{
    int r = 0;
    do {
        r++;
    } while ((y+r)*r <= x)
    return --r;
}
```

For example, on input `193` the program should output `13.892443`. Translate this program into MIPS assembly code.

3. Write a program which prompts the user for three positive numbers x, n and p, and outputs $x^n \bmod p$. Your program should use the recursive version of modular exponentiation:

$$x^n \bmod p = \begin{cases} 1, & n = 0 \\ (x^2 \bmod p)^{n/2}, & n \text{ even} \\ (x(x^2 \bmod p)^{\lfloor n/2 \rfloor}) \bmod p, & n \text{ odd} \end{cases}$$

MIPS Program Conventions

1. You should use consistent alignment for all your instructions. Labels should start in the first column. All instructions should be indented by a constant amount, and all arguments to the instructions should all start in the same column. All comments should also start in the same column.

2. You should liberally comment your code. Specifically, any MIPS instruction which completes the execution of a given HLL line should display the HLL line as a comment. In general, you should not have more than 2 or 3 lines in a row which are uncommented.

3. Your files should all start with a single `.text` segment followed by a single `.data` segment (if necessary).

4. Labels for branches and unconditional jumps in the text segment should be of the form `Ln`, where `n` = 1, 2, 3, etc. All labels should appear in numerical order, i.e. the first line needing a label should be labeled `L1`, the next `L2`, etc. Note that referencing these labels may not occur in numerical order (e.g., you may have a `bne ..., L2` before a `j L1`). Labels for jump-and-link instructions should use the name of the function. Labels in the data segment can be of your choosing.

5. Each of your programs should be labeled `pnxxx.asm` where `n` is either 1, 2 or 3 (corresponding to the problem being solved) and `xxx` is the first three letters of your last name. When you are finished, copy your programs to your `submit` folder or e-mail them to me.