Computer Organization

Assignment 4: Iterative Constructs

Sambit Sahoo (22277)

Q1 - MIPS assembly program that takes one number as an input, let say n and print the first n number of Fibonacci series by first storing into the memory.

```
#Sambit Sahoo 22277
.data
        mydetails: .asciiz "Sambit Sahoo 22277 \n\n"
        msg0: .asciiz "Enter the number n: "
        msg1: .asciiz " number of fibonacci series is stored at memory location "
        msg2: .asciiz "fibonacci series: "
        mem1: .byte 0
        msg3: .asciiz ", "
        msg4: .asciiz "\n"
.text
.globl main
main:
        li $v0, 4
        la $a0, mydetails
        syscall

        li $v0, 4
        la $a0, msg0
        syscall

        li $v0, 5
        syscall
        move $t0, $v0                   #t0 = n

        la $t2, mem1
        li $t3, 1
        sb, $t3, 4($t2)                 #memory[mem1 + 4] = 1

        #Fibonacci logic
        li $t4, 3                       #j = 1
        loop2:
                bgt $t4, $t0, break2

                lbu $t5, 0($t2)         #t5 = memory[mem1]
                lbu $t6, 4($t2)         #t6 = memory[mem1 + 4]
```

```
                add $t7, $t6, $t5
                sb $t7, 8($t2)          #memory[mem1 + 8] = t7

                addi $t4, $t4, 1        #j++

                addi $t2, $t2, 4        #mem1 = mem1 + 4
                j loop2
        break2:

        li $t1, 1                       #i = 1
        la $t2, mem1                    #again initializing
        loop1:
                bgt $t1, $t0, break1

                li $v0, 1
                move $a0, $t1
                syscall                 #prints 1, 2, 3......

                li $v0, 4
                la $a0, msg1
                syscall                 #prints "number of fibonacci series is stored at memory location"

                li $v0, 1
                move $a0, $t2           #prints the current $t2 or mem1
                syscall

                li $v0, 11
                li $a0, 10              #prints \n
                syscall

                addi $t2, $t2, 4        #updates by mem1 = mem1 + 4
                addi $t1, $t1, 1        #i++
                j loop1
        break1:
```

```
        li $v0, 11
        li $a0, 10              #prints \n
        syscall

        li $v0, 4
        la $a0, msg2
        syscall                 #prints "fibonacci series: "

        li $t1, 1
        la $t2, mem1
        addi $t0, $t0, -1
        loop3:

                lbu $t9, 0($t2)
                li $v0, 1
                move $a0, $t9
                syscall                 #prints the current value of $t2 or mem1

                bgt $t1, $t0, break3

                li $v0, 4
                la $a0, msg3
                syscall

                addi $t2, $t2, 4        #updates by mem1 = mem1 + 4
                addi $t1, $t1, 1        #i++
                j loop3
        break3:

        exit:
                li $v0, 10
                syscall
```

Console

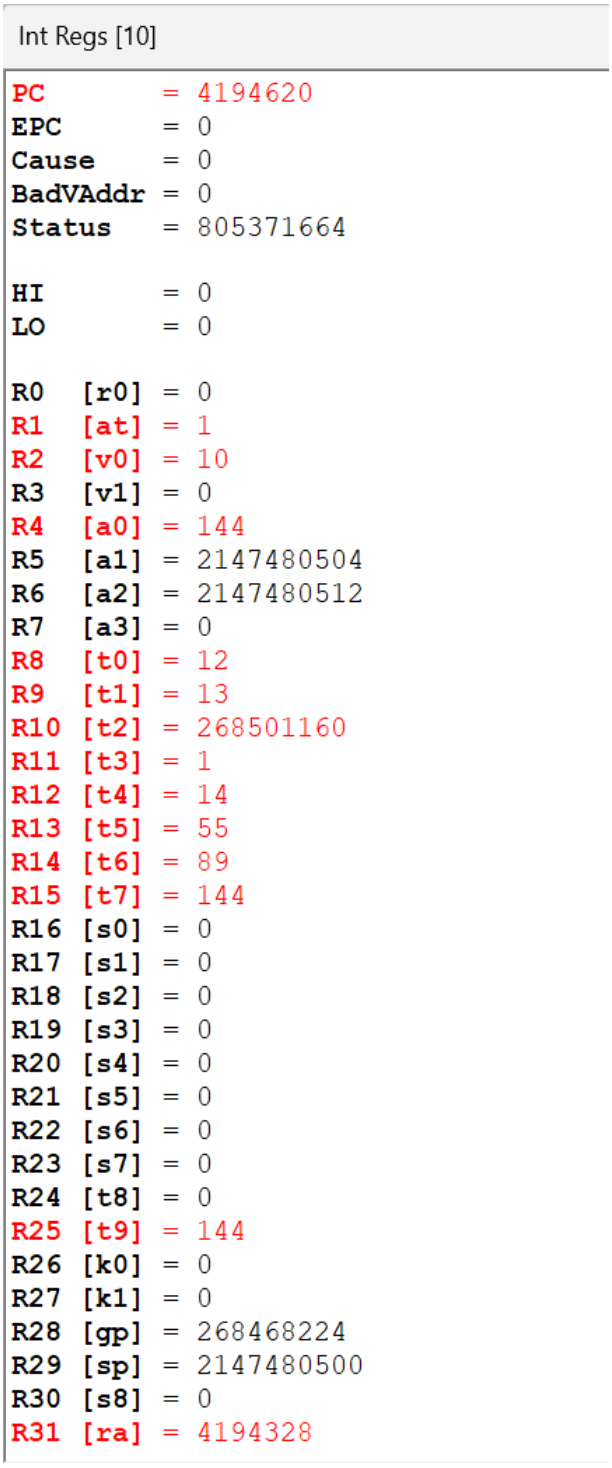
```
Sambit Sahoo 22277

Enter the number n: 13
1 number of fibonacci series is stored at memory location 268501112
2 number of fibonacci series is stored at memory location 268501116
3 number of fibonacci series is stored at memory location 268501120
4 number of fibonacci series is stored at memory location 268501124
5 number of fibonacci series is stored at memory location 268501128
6 number of fibonacci series is stored at memory location 268501132
7 number of fibonacci series is stored at memory location 268501136
8 number of fibonacci series is stored at memory location 268501140
9 number of fibonacci series is stored at memory location 268501144
10 number of fibonacci series is stored at memory location 268501148
11 number of fibonacci series is stored at memory location 268501152
12 number of fibonacci series is stored at memory location 268501156
13 number of fibonacci series is stored at memory location 268501160

fibonacci series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144
```

Double Precision

FP0	=	0.000000
FP2	=	0.000000
FP4	=	0.000000
FP6	=	0.000000
FP8	=	0.000000
FP10	=	0.000000
FP12	=	0.000000
FP14	=	0.000000
FP16	=	0.000000
FP18	=	0.000000
FP20	=	0.000000
FP22	=	0.000000
FP24	=	0.000000
FP26	=	0.000000
FP28	=	0.000000
FP30	=	0.000000

FP Regs

```
FIR     = 38912
FCSR    = 0


Single Precision
FG0  = 0.000000
FG1  = 0.000000
FG2  = 0.000000
FG3  = 0.000000
FG4  = 0.000000
FG5  = 0.000000
FG6  = 0.000000
FG7  = 0.000000
FG8  = 0.000000
FG9  = 0.000000
FG10 = 0.000000
FG11 = 0.000000
FG12 = 0.000000
FG13 = 0.000000
FG14 = 0.000000
FG15 = 0.000000
FG16 = 0.000000
FG17 = 0.000000
FG18 = 0.000000
FG19 = 0.000000
FG20 = 0.000000
FG21 = 0.000000
FG22 = 0.000000
FG23 = 0.000000
FG24 = 0.000000
FG25 = 0.000000
FG26 = 0.000000
FG27 = 0.000000
FG28 = 0.000000
FG29 = 0.000000
FG30 = 0.000000
FG31 = 0.000000
```



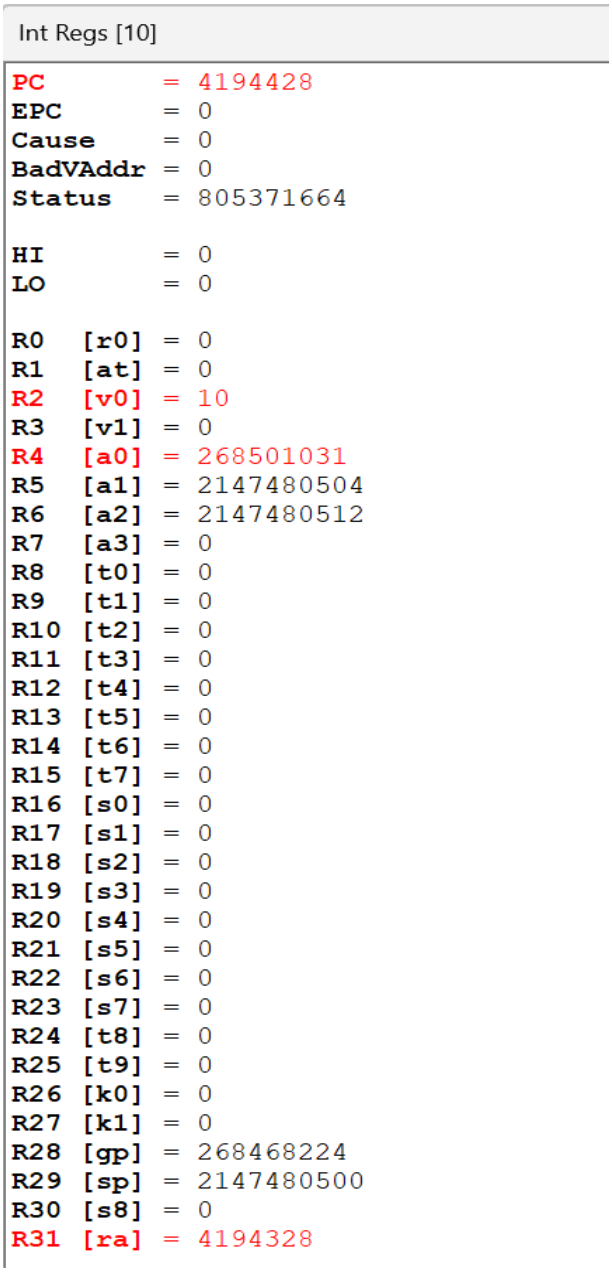
Q2 - Assembly program that lets the user input an integer and then prints a countdown from that number to 1 (inclusive), each value on a separate line. For example, if the user enters 4, the program should output the following sequence:

```
4
3
2
1
```

```
# Sambit Sahoo 22277

.data
        mydetails: .asciiz "Sambit Sahoo 22277 \n\n"
        msg1: .asciiz "Enter a number: "
        msg2: .asciiz "\n"


.text
.globl main
main:
        li $v0, 4
        la $a0, mydetails
        syscall

        li $v0, 4
        la $a0, msg1
        syscall

        li $v0, 5
        syscall
        move $t0, $v0           #t0 = n
```

```
        loop:
                ble $t0, $0, exit       #if n<=0

                li $v0, 1       #Print current number
                move $a0, $t0
                syscall

                li $v0, 4       #newline
                la $a0, msg2
                syscall

                addi $t0, $t0, -1       #decrement
                j loop

        exit:
                li $v0, 10
                syscall
```


Console
Sambit Sahoo 22277
Enter a number: 8
8
7
6
5
4
3
2
1


Int Regs [10]
PC = 4194428
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 268501031
R5 [a1] = 2147480504
R6 [a2] = 2147480512
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480500
R30 [s8] = 0
R31 [ra] = 4194328

FP Regs

```
FIR      = 38912
FCSR     = 0


Single Precision
FG0  = 0.000000
FG1  = 0.000000
FG2  = 0.000000
FG3  = 0.000000
FG4  = 0.000000
FG5  = 0.000000
FG6  = 0.000000
FG7  = 0.000000
FG8  = 0.000000
FG9  = 0.000000
FG10 = 0.000000
FG11 = 0.000000
FG12 = 0.000000
FG13 = 0.000000
FG14 = 0.000000
FG15 = 0.000000
FG16 = 0.000000
FG17 = 0.000000
FG18 = 0.000000
FG19 = 0.000000
FG20 = 0.000000
FG21 = 0.000000
FG22 = 0.000000
FG23 = 0.000000
FG24 = 0.000000
FG25 = 0.000000
FG26 = 0.000000
FG27 = 0.000000
FG28 = 0.000000
FG29 = 0.000000
FG30 = 0.000000
FG31 = 0.000000
Double Precision
FP0  = 0.000000
FP2  = 0.000000
FP4  = 0.000000
FP6  = 0.000000
FP8  = 0.000000
FP10 = 0.000000
FP12 = 0.000000
FP14 = 0.000000
FP16 = 0.000000
FP18 = 0.000000
FP20 = 0.000000
FP22 = 0.000000
FP24 = 0.000000
FP26 = 0.000000
FP28 = 0.000000
FP30 = 0.000000
```

Q3- Complete MIPS assembly program that implements the following C algorithm:

```
void main()
{
    int J = 1;
    int a;
    a = 0;

    while (1) {
        if (a > 20)
            break;
        J++;
        a += 2;
    }

    do {
        J++;
    } while (J < 100);

    while (a > 0) {
        J--;
        a--;
    }
}
```

```
# Sambit Sahoo 22277

.data
        mydetails: .asciiz "Sambit Sahoo 22277 \n\n"
        msg: .asciiz "$t0 = j \n$t1 = a"

.text
.globl main
main:
        li $v0, 4
        la $a0, mydetails
        syscall

        li $t0, 1       #j = 1
        li $t1, 0       #a = 0
        li $t2, 20      #t2 = 20
        loop1:
                bgt $t1, $t2, break1
                addi $t0, $t0, 1
                addi $t1, $t1, 2
                j loop1
        break1:
```

```
        loop2:
                addi $t0, $t0, 1
                li $t3, 100
                blt $t0, $t3, loop2

        break2:

        loop3:
                ble $t1, $0, break3
                addi $t0, $t0, -1
                addi $t1, $t1, -1
                j loop3
        break3:

        li $v0, 4
        la $a0, msg
        syscall

        exit:
                li $v0, 10
                syscall
```

Console

```
Sambit Sahoo 22277

$t0 = j
$t1 = a
```

```
Double Precision
FP0  = 0.000000
FP2  = 0.000000
FP4  = 0.000000
FP6  = 0.000000
FP8  = 0.000000
FP10 = 0.000000
FP12 = 0.000000
FP14 = 0.000000
FP16 = 0.000000
FP18 = 0.000000
FP20 = 0.000000
FP22 = 0.000000
FP24 = 0.000000
FP26 = 0.000000
FP28 = 0.000000
FP30 = 0.000000
```

FP Regs

```
FIR     = 38912
FCSR    = 0


Single Precision
FG0  = 0.000000
FG1  = 0.000000
FG2  = 0.000000
FG3  = 0.000000
FG4  = 0.000000
FG5  = 0.000000
FG6  = 0.000000
FG7  = 0.000000
FG8  = 0.000000
FG9  = 0.000000
FG10 = 0.000000
FG11 = 0.000000
FG12 = 0.000000
FG13 = 0.000000
FG14 = 0.000000
FG15 = 0.000000
FG16 = 0.000000
FG17 = 0.000000
FG18 = 0.000000
FG19 = 0.000000
FG20 = 0.000000
FG21 = 0.000000
FG22 = 0.000000
FG23 = 0.000000
FG24 = 0.000000
FG25 = 0.000000
FG26 = 0.000000
FG27 = 0.000000
FG28 = 0.000000
FG29 = 0.000000
FG30 = 0.000000
FG31 = 0.000000
```

Int Regs [10]

```
PC        = 4194440
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 805371664

HI        = 0
LO        = 0

R0  [r0] = 0
R1  [at] = 268500992
R2  [v0] = 10
R3  [v1] = 0
R4  [a0] = 268501014
R5  [a1] = 2147480504
R6  [a2] = 2147480512
R7  [a3] = 0
R8  [t0] = 78
R9  [t1] = 0
R10 [t2] = 20
R11 [t3] = 100
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480500
R30 [s8] = 0
R31 [ra] = 4194328
```

Q4- Write a MIPS assembly program that lets the user input an integer number k and prints the following pyramid pattern on the screen. For example, if the value of $k = 4$, the output should be:

```
   1
  121
 12321
1234321
```

```
# Sambit Sahoo 22277

.data
        mydetails: .asciiz "Sambit Sahoo 22277 \n\n"
        msg1: .asciiz "Give the input k = "
        printnewline: .asciiz "\n"


.text
.globl main
main:
        li $v0, 4
        la $a0, mydetails
        syscall

        li $v0, 4
        la $a0, msg1
        syscall

        li $v0, 5
        syscall
        move $t0, $v0   #t0 = n

        li $t1, 1                               #for i = 1


        loop1:
                bgt $t1, $t0, exit              #i > n
                sub $t2, $t0, $t1               #t2 = n - i
```

```
                li $t3, 1                       #for j = 1
                loop1a:
                        bgt $t3, $t2, break1a   #j > n - i

                        li $v0, 11
                        li $a0, 32              #prints space " "
                        syscall

                        addi $t3, $t3, 1        #t3++  (j++)
                        j loop1a
                break1a:


                li $t3, 1                       #for j = 1
                loop1b:
                        bgt $t3, $t1, break1b   #j <= i

                        li $v0, 1
                        move $a0, $t3
                        syscall                 #prints j

                        addi $t3, $t3, 1        #j++
                        j loop1b
                break1b:
```

```
                move $t3, $t1                   #for j = i
                li $t4, 2
                loop1c:
                        blt $t3, $t4, break1c   #j < 2

                        addi $t5, $t3, -1
                        li $v0, 1
                        move $a0, $t5
                        syscall                 #prints j-1

                        addi $t3, $t3, -1       #j--
                        j loop1c
                break1c:

                addi $t1, $t1, 1                #t1++  (i++)


                li $v0, 4
                la $a0, printnewline            #print newline
                syscall

                j loop1

        exit:
                li $v0, 10
                syscall
```


```
Double Precision
FP0  = 0.000000
FP2  = 0.000000
FP4  = 0.000000
FP6  = 0.000000
FP8  = 0.000000
FP10 = 0.000000
FP12 = 0.000000
FP14 = 0.000000
FP16 = 0.000000
FP18 = 0.000000
FP20 = 0.000000
FP22 = 0.000000
FP24 = 0.000000
FP26 = 0.000000
FP28 = 0.000000
FP30 = 0.000000
```

FP Regs

```
FIR      = 38912
FCSR     = 0


Single Precision
FG0  = 0.000000
FG1  = 0.000000
FG2  = 0.000000
FG3  = 0.000000
FG4  = 0.000000
FG5  = 0.000000
FG6  = 0.000000
FG7  = 0.000000
FG8  = 0.000000
FG9  = 0.000000
FG10 = 0.000000
FG11 = 0.000000
FG12 = 0.000000
FG13 = 0.000000
FG14 = 0.000000
FG15 = 0.000000
FG16 = 0.000000
FG17 = 0.000000
FG18 = 0.000000
FG19 = 0.000000
FG20 = 0.000000
FG21 = 0.000000
FG22 = 0.000000
FG23 = 0.000000
FG24 = 0.000000
FG25 = 0.000000
FG26 = 0.000000
FG27 = 0.000000
FG28 = 0.000000
FG29 = 0.000000
FG30 = 0.000000
FG31 = 0.000000
```

Int Regs [10]

```
PC        = 4194528
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 805371664

HI        = 0
LO        = 0

R0  [r0] = 0
R1  [at] = 1
R2  [v0] = 10
R3  [v1] = 0
R4  [a0] = 268501034
R5  [a1] = 2147480504
R6  [a2] = 2147480512
R7  [a3] = 0
R8  [t0] = 9
R9  [t1] = 10
R10 [t2] = 0
R11 [t3] = 1
R12 [t4] = 2
R13 [t5] = 1
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480500
R30 [s8] = 0
R31 [ra] = 4194328
```