OCTOBER 18, 2017

Tutorial 3

COMPUTER ARCHITECTURE

CS3021

MARK NOONE

15319898
MANOONE@TCD.IE

QUESTION 1

NON-OPTIMIZED

```
            add     r0, #4, r3

min:        add     r0, r26, r1
            sub     r27, r16, r0{C}
            jge     min1
            xor     r0, r0, r0
            add     r0, r27, r1

min1:       sub     r0, r28, r1
            jge     min2
            xor     r0, r0, r0
            add     r0, r28, r1

min2:       ret     r25, r0
            xor     r0, r0, r0


p:          add     r0, r3, r10
            add     r0, r26, r11
            add     r0, r27, r12
            callr   r25, min
            xor     r0, r0, r0
            add     r0, r1, r10
            add     r0, r28, r11
            add     r0, r29, r12
            callr   r25, min
            xor     r0, r0, r0
            ret     r25, r0
            xor     r0, r0, r0
```

```
gcd:    sub     r27, r0, r0{C}
        jeq     gcd1
        xor     r0, r0, r0
        add     r0, r26, r1
        ret     r25, r0
        xor     r0, r0, r0
gdc1:   add     r0, r27, r11
        add     r0, r26, r10
        callr   r25, mod
        xor     r0, r0, r0
        add     r0, r1, r11
        add     r0, r27, r10
        callr   r25, gcd
        xor     r0, r0, r0
        ret     r25, r0
        xor     r0, r0, r0
```

OPTIMIZED

```
        add     r0, #4, r3

min:    add     r0, r26, r1
        sub     r27, r16, r0{C}
        jge     min1
        xor     r0, r0, r0
        add     r0, r27, r1

min1:   sub     r0, r28, r1
        jge     min2
        xor     r0, r0, r0
        add     r0, r28, r1
```

```
min2:   ret     r25, r0
        xor     r0, r0, r0

p:      add     r0, r3, r10
        add     r0, r26, r11
        callr   r25, min
        add     r0, r27, r12
        add     r0, r1, r10
        add     r0, r28, r11
        callr   r25, min
        add     r0, r29, r12
        ret     r25, r0
        xor     r0, r0, r0

gcd:    sub     r27, r0, r0{C}
        jeq     gcd1
        xor     r0, r0, r0
        ret     r25, r0
        add     r0, r26, r1
gdc1:   add     r0, r27, r11
        callr   r25, mod
        add     r0, r26, r10
        add     r0, r1, r11
        callr   r25, gcd
        add     r0, r27, r10
        ret     r25, r0
        xor     r0, r0, r0
```

QUESTION 2 – ACKERMANN FUNCTION

The following information is based on an Ackermann function as per the following Ackermann(3, 6):

- 6 Register Sets:
  - Number of Procedure Calls – 172233
  - Maximum register window depth – 6
  - Number of register window overflows – 84884
  - Number of register windows underflows – 84885
- 8 Register Sets:
  - Number of Procedure Calls – 172233
  - Maximum register window depth – 8
  - Number of register window overflows – 83910
  - Number of register windows underflows – 83911
- 16 Register Sets:
  - Number of Procedure Calls – 172233
  - Maximum register window depth – 16
  - Number of register window overflows – 80141
  - Number of register windows underflows – 80142

QUESTION 3

The Ackermann function took .007 seconds to execute on my system. To time this I made a C version of the function to be executed inside a program (C was used from lecture notes recommendation). To then time the program, I used Linux's time command followed by the program to be timed. After running this several times, the mean was calculated giving the answer as seen above.