Student ID: _____

Family Name: _____

Other Name: _____

Desk: _____

Date: _____

Exam Mark: _____

In-Workshop Examination, WRKXX, Week W, Semester 2, 20YY

Computer Systems
COMP SCI 2000, 7081

Official Reading Time:	5 mins
Writing Time:	40 mins
Total Duration:	45 mins

Questions	**Time**	**Marks**
Answer all 6 questions	40 mins	40 marks 40 Total

Instructions for Candidates

- This is a closed book exam.
- Answer all questions in the spaces provided.
- Examination material must not be removed from the examination room.
- You must attend your enrolled workshop.
- A student ID card must be displayed at all times.
- No calculators or other electronics are permitted.
- Mobile phones must be turned off.
- Personal effects may be kept in a bag but, this must be placed on the floor.
- No talking or looking at other student's work.

Permitted Materials

- Foreign language paper dictionaries permitted.

DO NOT COMMENCE WRITING UNTIL INSTRUCTED TO DO SO

Question 1

Using only Nand gates, draw the logic circuit and write HDL code for the And gate. Clearly label the internal wires with the names used in your HDL.

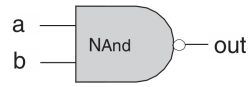
HDL

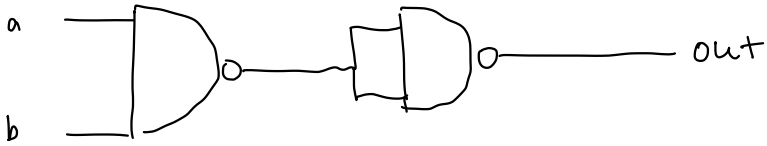
```
Chip And                                    // The available chips:
{                                           // Nand(a=?,b=?,out=?)
    IN a, b ;
    OUT out ;

    PARTS:
```

```
}
```

Circuit Diagram

Available chips:





[Total for Question 1: 4 marks]

Question 2

Using only Nand gates, draw the logic circuit and write HDL code for the Or gate. Clearly label the internal wires with the names used in your HDL.

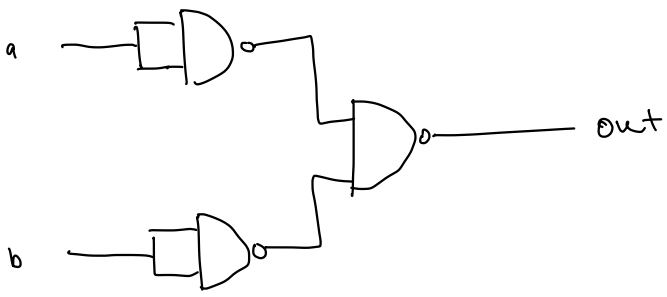
HDL

```
Chip Or                                    // The available chips:
{                                          // Nand(a=?,b=?,out=?)
    IN a, b ;
    OUT out ;

    PARTS:
```

```
}
```

Circuit Diagram

Available chips:



[Total for Question 2: 6 marks]

Question 3

Using only And, Or and Not gates, draw the logic circuit and write HDL code for the Xor gate. The Xor gate for two values a and b must be implemented as: $a.\overline{b} + \overline{a}.b$. Clearly label the internal wires with the names used in your HDL.

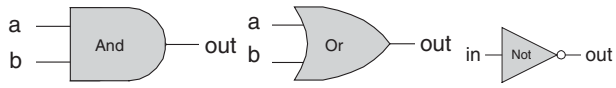
HDL

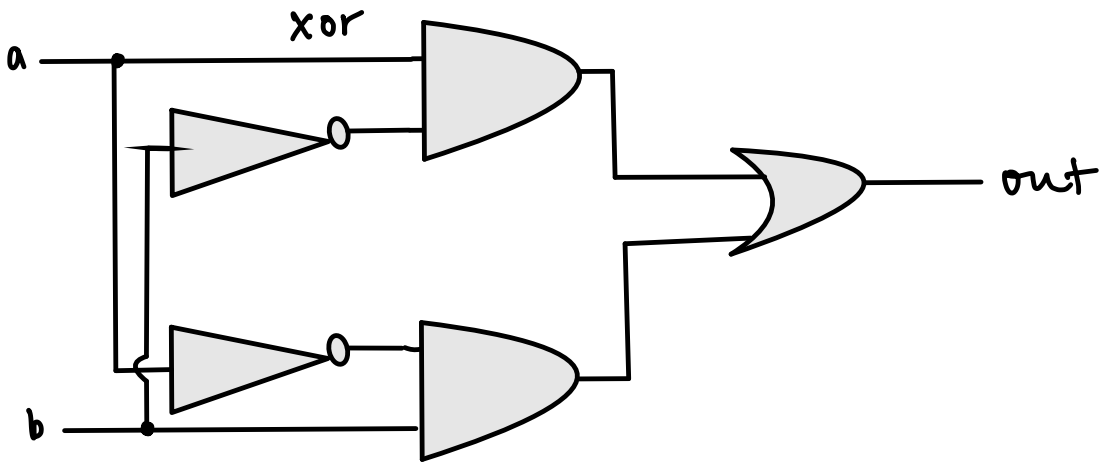
```
Chip Xor                                    // The available chips:
{                                           // And(a=?,b=?,out=?)
    IN a, b ;                               // Or(a=?,b=?,out=?)
    OUT out ;                               // Not(in=?,out=?)

    PARTS:
```

```
}
```

Circuit Diagram

Available chips:





[Total for Question 3: 10 marks]

Question 4

Using only And, Or and Not gates, draw the logic circuit and write HDL code for the Xnor gate. The Xnor gate for two values a and b must be implemented as: $\overline{a}.\overline{b} + a.b$. Clearly label the internal wires with the names used in your HDL.

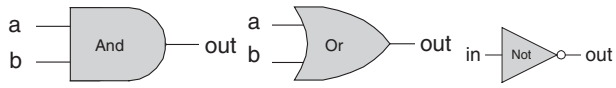
HDL

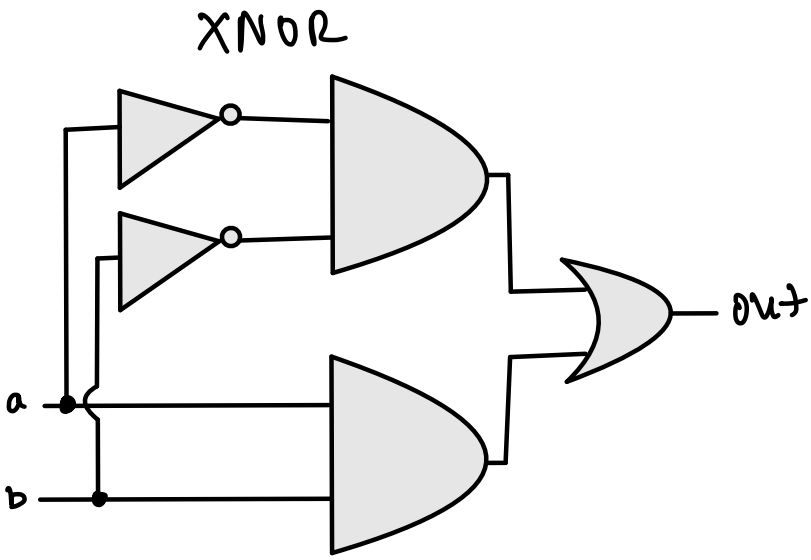
```
Chip Xnor                                   // The available chips:
{                                           // And(a=?,b=?,out=?)
    IN a, b ;                               // Or(a=?,b=?,out=?)
    OUT out ;                               // Not(in=?,out=?)

    PARTS:
```

```
}
```

Circuit Diagram

Available chips:





[Total for Question 4: 10 marks]

Question 5

Using only And, Or and Not gates, draw the logic circuit and write HDL code for the Mux gate. The Mux gate with two inputs **a**, **b** and a selector **sel** must be implemented as: $sel.b + \overline{sel}.a$. Clearly label the internal wires with the names used in your HDL.

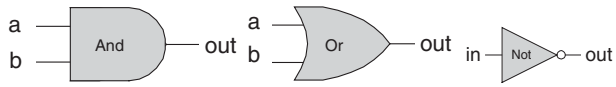
HDL

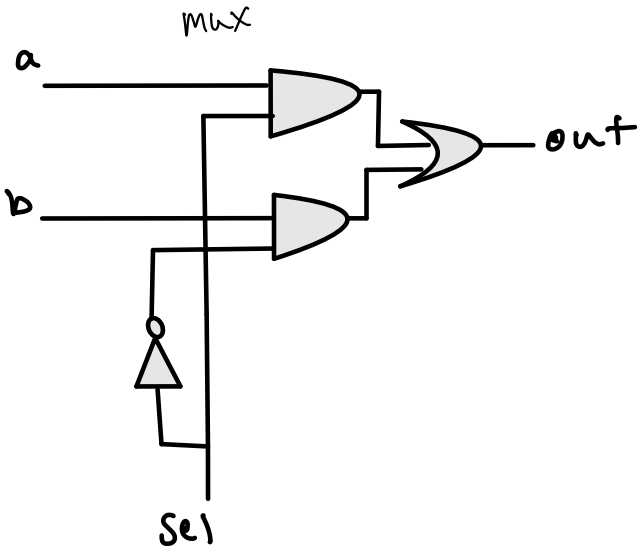
```
Chip Mux                                    // The available chips:
{                                           // And(a=?,b=?,out=?)
    IN a, b, sel;                           // Or(a=?,b=?,out=?)
    OUT out;                                // Not(in=?,out=?)

    PARTS:
```

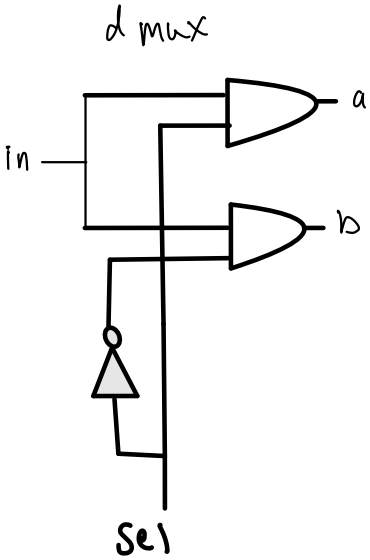
```
}
```

Circuit Diagram

Available chips:

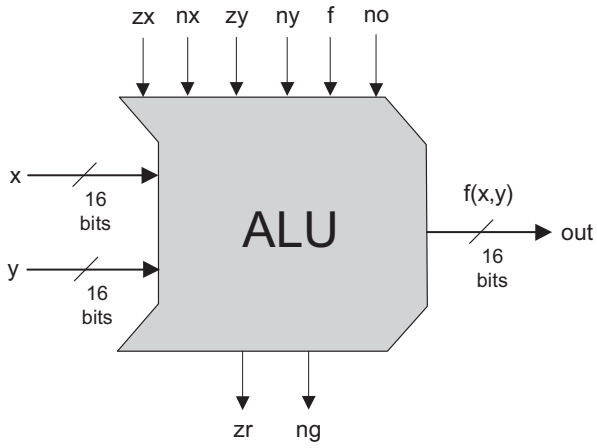






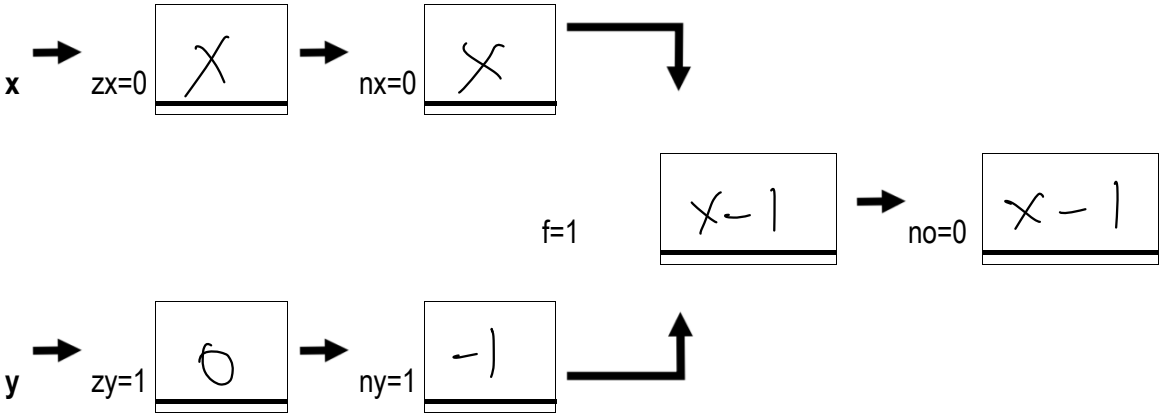
[Total for Question 5: 8 marks]

Question 6



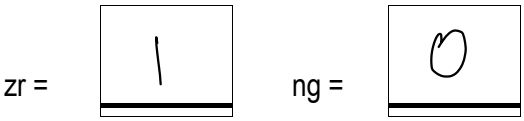
zx	nx	zy	ny	f	no
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x&y	if no then out=!out

a) Derive the function implemented by this ALU when the six control bits are as shown below:.



[6 marks]

b) Indicate the values of the outputs **zr** and **ng** if the values of **x** and **y** are 1 and 2 respectively.



[2 marks]

[Total for Question 3: 6 marks]

The following table shows the effect of 18 different combinations of control inputs to this ALU:

zx	**nx**	**zy**	**ny**	**f**	**no**
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x&y	if no then out=!out
0	**0**	**0**	**0**	**0**	**0**
x=x	x=x	y=y	y=y	out=x&y	out=x&y
0	**0**	**0**	**0**	**1**	**0**
x=x	x=x	y=y	y=y	out=x+y	out=x+y
0	**0**	**0**	**1**	**1**	**1**
x=x	x=x	y=y	y=-y-1	out=x-y-1	out=y-x
0	**0**	**1**	**1**	**0**	**0**
x=x	x=x	y=0	y=-1	out=x	out=x
0	**0**	**1**	**1**	**0**	**1**
x=x	x=x	y=0	y=-1	out=x	out=!x
0	**0**	**1**	**1**	**1**	**0**
x=x	x=x	y=0	y=-1	out=x-1	out=x-1
0	**0**	**1**	**1**	**1**	**1**
x=x	x=x	y=0	y=-1	out=x-1	out=-x
0	**1**	**0**	**0**	**1**	**1**
x=0	x=-x-1	y=y	y=y	out=y-x-1	out=x-y
0	**1**	**0**	**1**	**0**	**1**
x=x	x=!x	y=y	y=!y	out=!x&!y	out=x|y
0	**1**	**1**	**1**	**1**	**1**
x=x	x=-x-1	y=0	y=-1	out=-x-2	out=x+1
1	**0**	**1**	**0**	**1**	**0**
x=0	x=0	y=0	y=0	out=0	out=0
1	**1**	**0**	**0**	**0**	**0**
x=0	x=-1	y=y	y=y	out=y	out=y
1	**1**	**0**	**0**	**0**	**1**
x=0	x=-1	y=y	y=y	out=y	out=!y
1	**1**	**0**	**0**	**1**	**0**
x=0	x=-1	y=y	y=y	out=y-1	out=y-1
1	**1**	**0**	**0**	**1**	**1**
x=0	x=-1	y=y	y=y	out=y-1	out=-y
1	**1**	**0**	**1**	**1**	**1**
x=0	x=-1	y=y	y=-y-1	out=-y-2	out=y+1
1	**1**	**1**	**0**	**1**	**0**
x=0	x=-1	y=0	y=0	out=-1	out=-1
1	**1**	**1**	**1**	**1**	**1**
x=0	x=-1	y=0	y=-1	out=-2	out=1

Question 7

What is the largest decimal number that can be represented by a two's complement 16-bit binary number?

Answer: 32767

$2^{15} - 1$

[Total for Question 7: 2 marks]

Question 8

What is the largest decimal number that can be represented by a 16-bit unsigned binary number?

Answer: 65535

[Total for Question 8: 2 marks]

Question 9

What is the most negative decimal number that can be represented by a two's complement 16-bit binary number?

Answer: -32768

-2^{k-1}

[Total for Question 9: 2 marks]

Question 10

What is the smallest decimal number that can be represented by an unsigned 16-bit binary number?

Answer: 0

[Total for Question 10: 2 marks]

Question 11

What is the decimal value of the 8-bit two's complement number 11101010_2?

Answer: -22

0 0 0 1 0 1 0 1
0 0 0 1 0 1 1 0
16 4 2
-22

[Total for Question 11: 2 marks]

Question 12

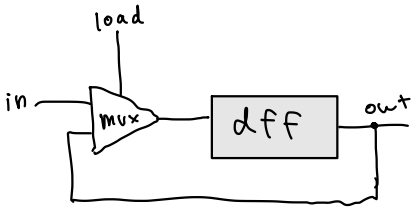
Draw the logic circuit and write the HDL implementation for a 1-bit register using Mux and DFF chips. Clearly label the chips and internal wires with the names used in your HDL.

HDL

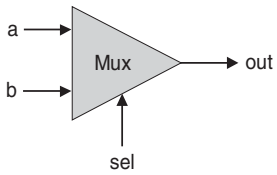
```
Chip Bit                                    // The available chips:
{                                           // DFF(in=?,out=?)
    IN in, load ;                           // Mux(a=?,b=?,sel=?,out=?)
    OUT out ;

    PARTS:
```

```
}
```

Circuit Diagram



Available chips:



[Total for Question 12: 4 marks]

Question 13

Draw the logic circuit and write the HDL implementation for a HalfAdder using Xor and And chips. Clearly label the chips and internal wires with the names used in your HDL.

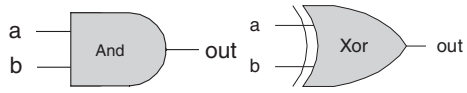
HDL

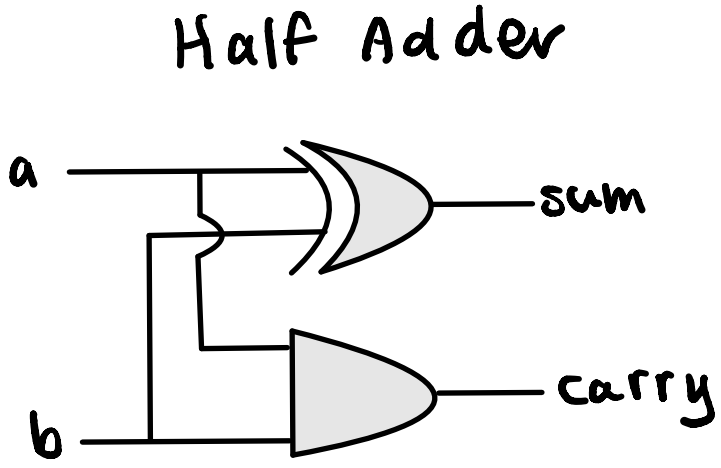
```
Chip HalfAdder                          // The available chips:
{                                       // Xor(a=?,b=?,out=?)
    IN a, b ;                           // And(a=?,b=?,out=?)
    OUT sum, carry ;

    PARTS:
```

}

Circuit Diagram

Available chips:





[Total for Question 13: 4 marks]

Question 14

Draw the logic circuit and write the HDL implementation for a FullAdder using HalfAdder and Or chips. Clearly label the chips and internal wires with the names used in your HDL.

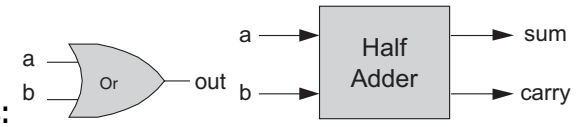
HDL

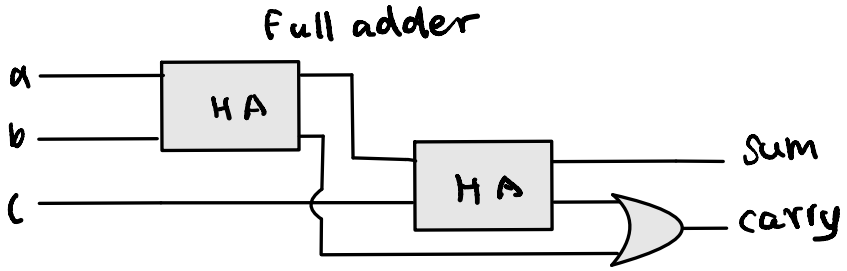
```
Chip FullAdder                          // The available chips:
{                                       // HalfAdder(a=?,b=?,sum=?,carry=?)
    IN a, b, c ;                        // Or(a=?,b=?,out=?)
    OUT sum, carry ;

    PARTS:
```

```
}
```

Circuit Diagram

Available chips:





[Total for Question 14: 4 marks]

Question 15

Draw the logic circuit and write HDL code for a RAM with 2 registers using Register, Mux16, and Dmux chips. Clearly label the chips and internal wires with the names used in your HDL.

HDL

```
Chip RAM                              // The available chips:
{                                     // Dmux (in=?,sel=?,a=?,b=?)
    IN in[16], load, address ;        // Register(in=?,load=?,out=?)
    OUT out[16] ;                     // Mux16(a=?,b=?,sel=?,out=?)

    PARTS:
```

```
}
```

Circuit Diagram

[Total for Question 15: 12 marks]

Question 16

Draw the logic circuit and write the HDL for the Hack machine's program counter, PC. Clearly label the chips and internal wires with the names used in your HDL.

HDL

```
Chip PC                              // The available chips:
{                                    // Register(in=?,load=?,out=?)
    IN in[16], load, inc, reset;     // Mux16(a=?,b=?,sel=?,out=?)
    OUT out[16] ;                    // Inc16(in=?,out=?)
    PARTS:
```

```
}
```

Circuit Diagram

```
Function: If reset(t-1) then out(t)=0
             else if load(t-1) then out(t)=in(t-1)
                    else if inc(t-1) then out(t)=out(t-1)+1
                           else out(t)=out(t-1)
```

[Total for Question 16: 12 marks]

Question 17

What three kinds of symbols can appear in a Hack Assembly Language program?

1. Predefined
2. labels
3. variables

[Total for Question 17: 6 marks]

Question 18

When are the values determined for the three kinds of symbols that can appear in Hack Assembly?

1. initialisations
2. first pass
3. second pass

[Total for Question 7: 18 marks]

Question 19

Excluding predefined symbols, show the symbol table for the following Hack Assembly at the end of the first pass:

```
@R0       0
D=M       1
@END      2
D;JLE     3
@counter  4
M=D       5
@x        6
M=D       7
(LOOP)
D=D+A     8
@LOOP     9
D;JGT     10
(END)
@END      11
0;JMP     12
```

Symbol	Value
LOOP	8
END	11

[Total for Question 19: 4 marks]

Question 20

Show the final symbol table entries for the variables in the following Hack Assembly program:

```
@R0
D=M
@END
D;JLE
@counter
M=D
@x
M=D
(LOOP)
D=D+A
@LOOP
D;JGT
(END)
@END
0;JMP
```

Symbol	Value
counter	16
x	17

[Total for Question 20: 4 marks]

Question 21

Implement the following code fragments in Hack Virtual Machine code, the variables ***a***, ***b*** and ***c*** are in the local segment at offsets 4, 5 and 6 respectively.

Jack Code **Virtual Machine Code**

a) ~ (a | b)

```
push local 4
push local 5
or
not
```

[4 marks]

b) (a + (b + c))

```
push local 4
push local 5
push local 6
add
add
```

[5 marks]

c) ((a + b) + c)

```
push local 4
push local 5
add
push local 6
add
```

[5 marks]

d) Recursive.factorial(6)

```
push constant 6
call Recursive.factorial 1
```

[2 marks]

e) a = c * b

```
push local 6
push local 5
call Math.multiply 2
pop local 4
```

[4 marks]

[Total for Question 21: 20 marks]

Question 22

Implement the following code fragments in Hack Virtual Machine code, the variables ***a***, ***b*** and ***c*** are in the local segment at offsets 4, 5 and 6 respectively.

Jack Code **Virtual Machine Code**

a) a = 93

```
push constant 93
```

[2 marks]

b) Math.multiply(b,c)

```
push local 5
push local 6
call math.multiply 2
```

[3 marks]

c) return 17

```
push constant 17
return
```

[2 marks]

[Total for Question 22: 7 marks]

Question 23

Complete the Hack Virtual Machine code that implements the body of the following Jack function:

Jack Code

```
function add(int x,int y)
{
        int sum ;
        let sum = x + y ;


        return sum ;

}
```

Virtual Machine Code

```
function Useful.add 1
push arg 0
push arg 1
add
pop local 0
push local 0


return
```

[Total for Question 23: 5 marks]

Question 24

Complete the Hack Virtual Machine code that implements the body of the following Jack function:

Jack Code

```
function nfib(int n)
{
    if ( n <= 1 )
    {
        return 1 ;
    }
    return (1 + nfib(n-1)) + nfib(n-2) ;
}
```

Virtual Machine Code

```
function Useful.nfib 0
push argument 0
push constant 1
le
not
if-goto if_false
push constant 1
return
label if_false
push constant 1
push argument 0
push constant 1
sub
call Useful.fib 1
add
push argument 0
push constant 2
sub
call Useful.fib 1
add
return
```

[Total for Question 24: 15 marks]

Question 25

Complete the Hack Virtual Machine code that implements the body of the following Jack function:

Jack Code

```
function triangle(int n)
{
    If ( n <= 1 )
    {
        return 1 ;
    }
    return n + Useful.triangle(n – 1) ;
}
```

Virtual Machine Code

```
function Useful.triangle 1
push arg 0
push constant 1
le
not

if-goto if_false
push constant 1

return

label if_false
push argument 0
push argument 0
push constant 1
sub
call Useful.triangle 1
add

return
```

[Total for Question 25: 10 marks]

Question 26

Write Hack Assembly Language that will implement the following Hack Virtual Machine commands:

Virtual Machine Code **Assembly Language Code**

a) push constant 0

```
@ SP
A M = M+1
A = A - 1
M = 0
```

[5 marks]

b) pop local 1

```
@ LCL
D = M
@ 1
D = D + A
@ 16
M = D
```

```
@ SP
A M = M-1
D = M
@ 16
A = M
M = D
```

[6 marks]

c) push argument 56

```
@ ARG
D = M
@ 56
D = D + A
A = D
D = M
@ SP
AM = M+1
A = A-1
M = D
```

[10 marks]

[Total for Question 26: 21 marks]

Question 27

Write Hack Assembly Language that will implement the following Hack Virtual Machine commands:

Virtual Machine Code **Assembly Language Code**

d) add

```
@SP
AM = M-1
D = M
A = A-1
M = M+D
```

[5 marks]

d) sub

```
AM = M-1
D = M
A = A-1
M = M-D
```

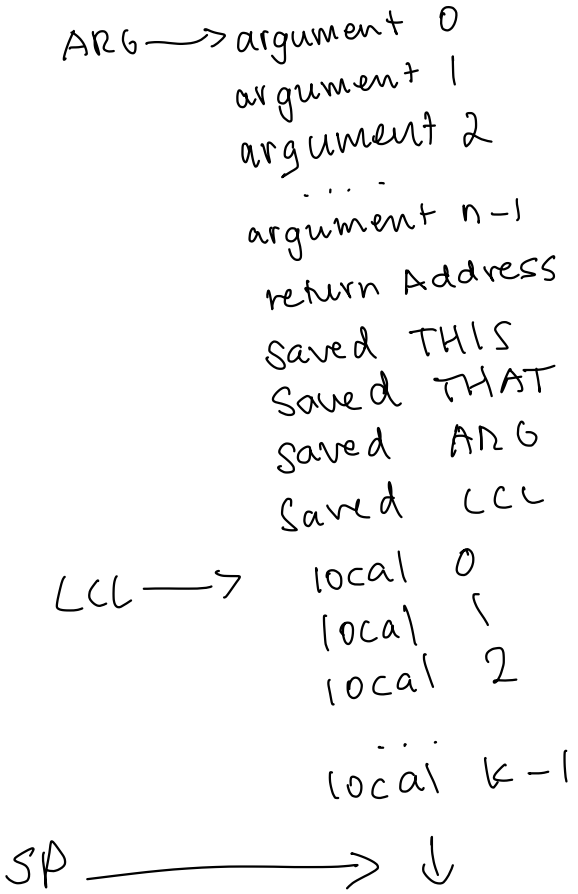
[5 marks]

[Total for Question 27: 10 marks]

Question 28

Draw the stack frame of the currently executing function in the Hack Virtual machine. It was passed ***n*** arguments and has ***k*** local variables, ***n*** and ***k*** are both greater than 3. Your answer must show where the ARG, LCL and SP virtual registers are pointing.

Diagram



[Total for Question 28: 12 marks]

End of Examination