CSC 106, Summer 2016, Assignment #2
Due Wednesday, June 15 at 11:55pm
100 marks total, worth 6% of your final grade

Question 1: Algorithm Efficiency [30 Marks]

When we wrote the flowchart for identifying the poisoned bottle in class (Lecture 6), we replaced:

raisedBase = 2^i

by

raisedBase = *1* as a base case (2^0)

raisedBase = *raisedBase* + *raisedBase* for all other cases

to avoid computation redundancy.

Assume that:

raisedBase = *1* counts as one operation

adding two numbers counts as one operation

multiplying two numbers counts as one operation

calculating exponentials implies multiplying the base to itself as many times as the exponent specifies (except for the base case where $2^0 = 1$).

Eg: $2^3 = 2x2x2$ and $2^0 = 1$

a) (5 marks) Given that our binary number *deathBin* has length 10 (10 bits), how many operations do we need to perform to compute *raisedBase* from the start to the end of the algorithm? Explain. (note: We are only interested in the operations related to the processes starting with *raisedBase* = …).

b) (10 marks) How many operations would we need to perform to compute *raisedBase,* from the start to the end of the algorithm (with the same input), if we choose to use *raisedBase* = 2^i instead? Include your calculations.

c) (5 marks) Imagine that we use our algorithm to translate a 64-bit binary number to decimal. How many operations would we need to compute *raisedBase,* from the start to the end of the algorithm, if we used our algorithm? Explain.

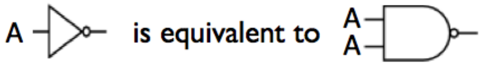
d) (10 marks) With a 64-bit binary number, how many operations would we need to compute *raisedBase,* from the start to the end of the algorithm, if we choose to use *raisedBase* = 2^i instead? Include your calculations.

Q2: Logic [30 marks]

Recall NAND gates from your lab, with truth table:

A	B	¬(A∧B)
0	0	1
0	1	1
1	0	1
1	1	0

Any Boolean function can be implemented using combinations of NAND gates. This means that you can build an entire processor using only NAND gates. For example, you can replace a NOT gate by a NAND gate by sending the same input twice into the gate:



a) (20 marks) Implement the following Boolean formula using only NAND gates:

$$(A \wedge B) \wedge (\neg C \vee A)$$

b) (10 marks) Replace and XOR gate by a combination of NAND gates:

Q3: Sorting Algorithms [40 marks]

Two of the sorting algorithms we have seen in class and labs are: Insertion Sort and Selection Sort. Given the following pseudocodes, and input A = [1,6,2,8,3,4]:

a) (20 marks) What is the printed output for this Insertion Sort pseudocode:

```
insertionSort(A)
      for i = 1 to (length of A)-1
            temp = A[i]
            j = i-1
            while j >= 0 and A[j] > temp
                  A[j+1] = A[j]
                  print "swapping" and current state of A
                  j = j-1
            A[j+1] = temp
            print current state of A
```

b) (20 marks) What is the printed output for this Selection Sort pseudocode:

```
selectionSort(A)
      for i = 0 to (length of A)-1
            minPos = i
            j = i+1
            while j < length of A
                  if A[j] < A[minPos]
                        minPos = j
                  j = j+1
            if minPos is not = i
                  temp = A[i]
                  A[i] = A[minPos]
                  A[minPos] = temp
                  print "swapping" and current state of A
            i = i+1
            print current state of A
```

Final notes

Submit your assignment as a **PDF** through the connex site. Any submission in other format than **PDF** will be penalized.

No late assignment accepted.