University of Waterloo
CS240 - Fall 2015
Assignment 1

Due Date: Wednesday September 30 at 5:00pm

Please read `http://www.student.cs.uwaterloo.ca/~cs240/f15/guidelines.pdf` for guidelines on submission. Problems 1 – 8(a) are written problems; submit your solutions electronically as a PDF with file name `a01wp.pdf` using MarkUs. We will also accept individual question files named `a01q1w.pdf`, `a01q2w.pdf`, ... , `a01q8w.pdf` if you wish to submit questions as you complete them. Submit your solution to 8(b) electronically as a file named `report.cpp`.

There are 73 marks available; the assignment will be marked out of 70.

Problem 1 [3+3+3+3+3=15 marks]

Provide a complete proof of the following statements from first principles (i.e., using the original definitions of order notation). All logarithms are natural logarithms: log = ln.

a) $12n^3 + 11n^2 + 10 \in O(n^3)$

b) $12n^3 + 11n^2 + 10 \in \Omega(n^3)$

c) $12n^3 + 11n^2 + 10 \in \Theta(n^3)$

d) $1000n \in o(n \log n)$

e) $n^n \in \omega(n^{20})$

Problem 2 [4+4=8 marks]

For each pair of the following functions, fill in the correct asymptotic notation among Θ, o, and ω in the statement $f(n) \in \sqcup(g(n))$. Provide a brief justification of your answers. In your justification you may use any relationship or technique that is described in class.

a) $f(n) = \sqrt{n}$ versus $g(n) = (\log n)^2$

b) $f(n) = n^3(5 + 2\cos 2n)$ versus $g(n) = 3n^2 + 4n^3 + 5n$

Problem 3 [6+6=12 marks]

Prove or disprove each of the following statements. To prove a statement, you should provide a formal proof that is based on the definitions of the order notations. To disprove a statement, you can either provide a counter example and explain it or provide a formal proof. All functions are positive functions.

a) $f(n) \notin o(g(n))$ and $f(n) \notin \omega(g(n)) \Rightarrow f(n) \in \Theta(g(n))$

b) $\min(f(n), g(n)) \in \Theta\left(\frac{f(n)g(n)}{f(n)+g(n)}\right)$

Problem 4 [4 marks]

Derive a closed form for the following sum:

$$S(n) = \sum_{i=1}^{n} i/2^i.$$

Problem 5 [2+2+4+4=12 marks]

Consider the following procedure.

```
pre: n is a positive integer
pre: v[1..n] is a binary vector of length n,
     i.e., each entry is either 0 or 1
foo(v,n)
     i := 1;
     while i<=n and v[i]=0 do
         i := i+1
     od;
     for j from 1 to i do
         print("Hello world!")
     od;
```

a) How many inputs are there are of size n?

b) What is the worst case number of calls to print? Give an exact formula in terms of n and justify your answer by giving an example of a worst case input of size n.

c) For $i \in \{1, 2, \ldots, n\}$, let S_i denote the subset of inputs of size n for which the number of calls to print is i. Describe and enumerate S_i.

d) What is the average case number of calls to print? Derive an exact closed form formula in terms of n.

Problem 6 [5 marks]

Prove that the following code fragment will always terminate.

```
s := 3*n  // n is an integer
while (s>0)
   if (s is even)
      s := floor(s/4)
   else
      s := 2*s
```

Problem 7 [5 marks]

Analyze the following piece of pseudo-code and give a tight bound (i.e. Θ notation) on the running time as a function of n. Show your work. A formal proof is not required, but you should justify your answer.

```
1.  $mystery \leftarrow 0$
2.  for $i \leftarrow 1$ to $3n$ do
3.      $mystery \leftarrow mystery \times 4$
4.      for $j \leftarrow 1388$ to $2010$ do
5.          for $k \leftarrow 4i$ to $6i$ do
6.              $mystery \leftarrow mystery + k$
```

Problem 8 [6+6=12 marks]

You are given an array $A[0 \ldots n-1]$ of integers (not necessarily distinct) that forms a max-heap of size n.

a) Describe an algorithm that takes as input an integer c, not necessarily in the heap, and reports all integers in the heap that are greater than or equal to c. The running time of your algorithm should be $O(k)$, where k is the number of integers reported/outputted, **not** the size of the input. Provide a brief explanation for why the running time of your algorithm is $O(k)$.

Please note that reading in the input into an **array** does not count towards your total running time, the array is already a max-heap. Inserting the input into a linked-list binary tree or binary heap is **incorrect**.

b) Implement your algorithm from part (a). Your program should read from `cin` the size n, then the n integers in the heap $A[0 \ldots n-1]$, and finally the integer c, and then write to `cout` the integers in the heap that are greater than or equal to c. You may assume that every integer in the input is at least 0 and at most $2^{31} - 1$ (so every integer will fit into a variable of type `int`).

Every integer in the input and output should be on a separate line. So for instance if the input consists of the following lines:

```
5
17
15
13
10
3
12
```

then your program should print out the integers `17`, `15`, and `13` in any order (and on separate lines).

Submit the code for your `main` function, along with any helper functions, in a file called `report.cpp`.