Computer Science Foundation Exam

May 3, 2002

COMPUTER SCIENCE I

Section I A

No Calculators!

KEY

Name: _____

SSN: _____

In this section of the exam, there are four (4) problems

You must do all of them.

The weight of each problem in this section is indicated with the problem. The algorithms in this exam are written in C programming language notation. Partial credit cannot be given unless all work is shown.

As always, be complete, yet concise, and above all be neat. Credit cannot be given when your results are unreadable.

(1, 16%) (C code) Given the following array and program segment, answer the questions below.

Array x	**5**	**2**	**6**	**1**	**3**	**7**
position	0	1	2	3	4	5

```
int x[6] = {5,2,6,1,3,7};
int i, j, k, a, b, t;

i=0; j=5; a=0; b=0;
while (i<j){
  if (x[i]<x[j]) {
    k=i+1;
    while (k<j) {
      if (x[k]<x[k+1]) {
        a=a+1;
        t=x[k];
        x[k]=x[k+1];
        x[k+1]=t;
      }
      k=k+1;
    }     // end of inner while loop
    x[i]=x[i]+a;
    i=i+1;
  }
  else {
    k=j-1;
    while (k>i) {
      if (x[k]>x[k-1]) {
        b=b+1;
        t=x[k-1];
        x[k-1]=x[k];
        x[k]=t;
      }
      k=k-1;
    }     // end of inner while loop
    x[j]=x[j]+b;
    j=j-1;
  }
} // end of outer while loop
```

a) Show the array **x** after the program segment is executed (2 pts. each).

Array x	**8**	**7**	**9**	**5**	**6**	**4**
position	0	1	2	3	4	5

b) What value will the variables **a** and **b** contain when the algorithm is finished (2 pts. each).

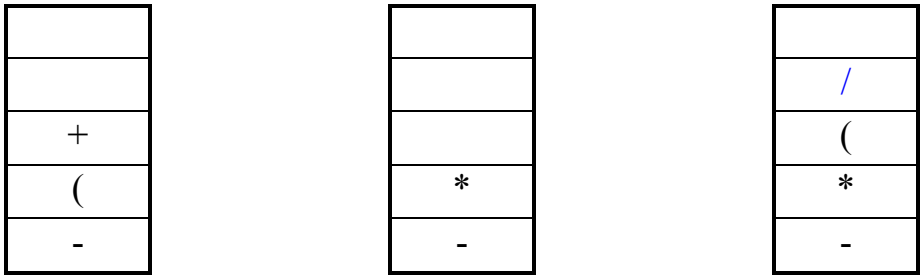
a	**3**	**b**	**3**

(2, 12%) Consider the following expression with the infix notation:

$$X * Y - (P + Q\ ^{A}) *\ ^{B}(R / S\ ^{C})$$

Transform this into a postfix expression. Trace the state of the operator stack as each character of the infix expression is processed. Show the contents of the operator stack at the indicated points in the infix expression (points A, B and C). Put the final postfix expression in the blank.

(3 pts. each)



A B C

Resulting Postfix Expression (3 pts): **XY*PQ+RS/*-**

(3, 14%) Assume that you have a stack S, a queue Q, and the standard stack - queue operations: `push, pop, enqueue and dequeue`. Assume that `print` is a function that prints the value of its argument. Execute, in top-to-bottom order, the operations below and answer questions **a)**, **b)** and **c)** below.

```
enqueue(Q,5);
push(S,4);
enqueue(Q,8);
push(S, 6);
push(S, dequeue(Q));
print(pop(S));
push(S, 7);
enqueue(Q,3);
enqueue(Q, pop(S));
print(dequeue(Q));
push(S, dequeue(Q));
enqueue(Q, 2);
push(S,5);
enqueue(Q, pop(S));
print(dequeue(Q));
print(pop(S));
print(dequeue(Q));
```

a) Show the output from the print statements (2 pts. each):

5	8	7	3	2
first output	second output	third output	fourth output	fifth output

b) After the above operations are completed, how many items are left in stack S?

Answer (2 pts.): **2**

c) After the above operations are completed, how many items are left in queue Q?

Answer (2 pts): **1**

(4, 12%) Answer each of the following "timing" questions concerning an algorithm of a particular order and a data set of a particular size. Assume that the run time is affected only by the size of the data set and not its composition and that **n** is an arbitrary integer. (3 pts. each)

a) Algorithm A runs in $O(n^3)$ time. For an input size of 10, the algorithm runs in 7 milliseconds. For another input size, the algorithm takes 189 milliseconds. What was that input size?

30

b) For an $O(n^k)$ algorithm, where k is a positive rational number, a friend tells you that instance of size m took 16 seconds to run. You run an instance of size 4m and find that it takes 256 seconds to run. What is the value of k?

k = 2

c) Algorithm A runs in $O(n^3)$ time and Algorithm B solves the same problem in $O(n^2)$ time. If algorithm A takes 5 milliseconds to complete for an input size of 10, and algorithm B takes 20 milliseconds for an input size of 10, what is the input size that you expect the two algorithms to perform about the same?

n = 40

d) Consider the following segment of C code:

```
x = 0;
for (k = 1; k<=n; k++){
    j=n;
    while (j >= 1){
        x = x + j;
        j = j/2;
    }
}
```

What is the order of the time efficiency of this program segment for an arbitrary n?

O(nlogn)