CMPS 212 Midterm- Fall 2005

Duration: 75 minutes

Name:_____________________ Username:______

1. (4 points) What is the height of a full binary tree in terms of N where N is the number of nodes?

2. (4 points) Order the following from fastest (least running time) to slowest:
$O(n^2)$, $O(n)$, $O(\log n)$, $O(n^3)$, $O(n \log n)$, $O(n^2 \log n)$.

3. (4 points) What is big-O for the following pseudocode:

```
for (int i = 0; i < n; ++i) {
      if (condition) {
            foo(); // O(n^2)
      }
      else {
            bar();// O(n)
      }
}
```

4. (4 points) In Java, how many bits are there in a char type?

5. (5 points) Given the following Java class:

```
public class Person
{
        int age;
        String name;
        float weight;
}
```

How would you best represent the above class in C? Assume that the name attribute can be no longer than 100 characters.

6. (9 points) Consider a binary tree that is built by inserting the sequence 4, 9, 2, 10, 3, 5, 8, 1 into an initially empty binary tree.

(a) Draw the tree constructed by this insertion seque nce:

(b) Given all possibilities of the insertion pattern, what is the maximum height tree that could be built?

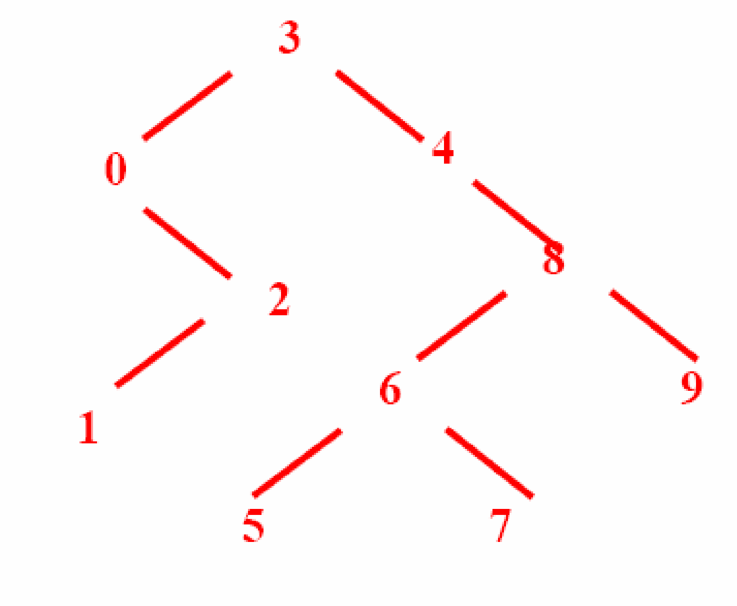
(c) Given all possibilities of the insertion pattern, what is the minimum height tree that could be built?

7) (4 points) Is it possible for an $O(n^3)$ algorithm to run faster than an $O(n)$ algorithm on a given input? True or False? Explain or give an example.

8. (6 points) Given the following infix expression: ((k + j) * (a - g)) / (e + b)
a) Write it as a prefix expression:

b) Write it as a postfix expression:

9. (8 points) Delete the root of the BST shown below using both of the two methods described in class.



Draw the 2 new trees here:

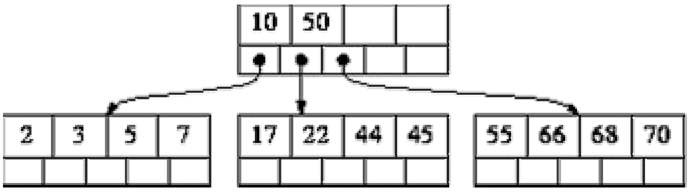
10. (8 points) Draw the contents of the hash table after inserting 5,16,25,33,41 given:
a) Table size is 9
b) Linear probing
c) h(k) = k mod tablesize

0	
1	
2	
3	
4	
5	
6	
7	
8	

11. (8 points) Draw the contents of the hash table after inserting 33,10,12,47,49,58 given:
a) Table size is 7
b) Quadratic probing
c) h(k) = k mod tablesize
(Hint: in quadratic probing the step size = probe*probe)

0	
1	
2	
3	
4	
5	
6	

12. (5 pts) Show the state of the following B-Tree after inserting 6 in it.



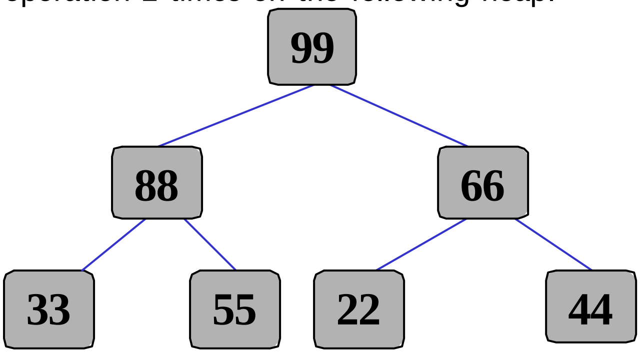
13. a) (5 pts) Take the following numbers and insert them into a heap in the given order. Show the content of the heap array after you perform this operation:

33 63 22 54 91 83 25

a[0] a[1] a[2] a[3] a[4] a[5] a[6] a[7]

| | | | | | | | | | | |

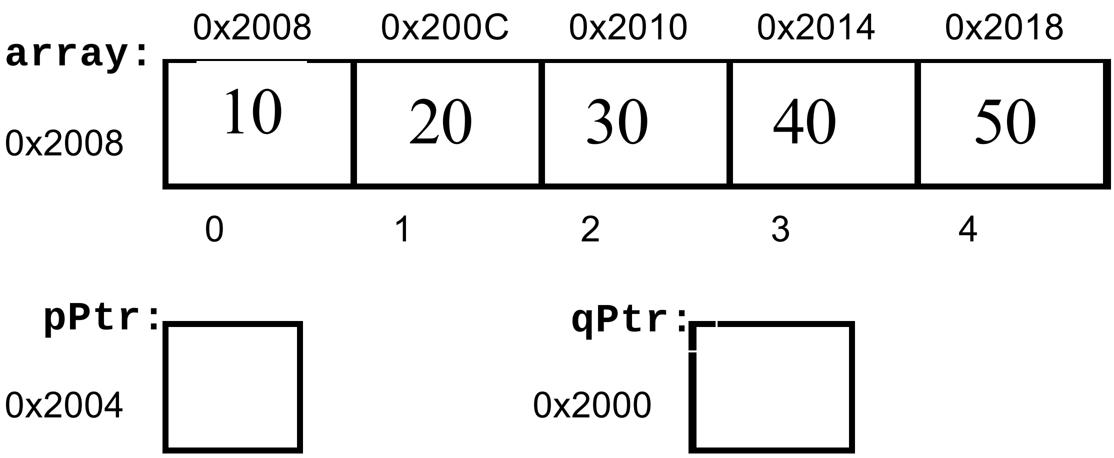
b) (5 pts) Show the content of the heap array after you perform the remove operation 2 times on the following heap.



a[0] a[1] a[2] a[3] a[4] a[5] a[6] a[7]

| | | | | | | | | | | |

14. (15 points) What does each of the print statements below display? (Numbers starting with 0x represent memory addresses)




```
float  array[5] = {10, 20, 30, 40, 50};

float* pPtr = array;
float* qPtr = NULL;

printf("%f\n", *pPtr);          →

printf("%u\n", &pPtr);          →

printf("%d\n", pPtr);           →

printf("%u\n", array) ;         →

printf("%d\n", qPtr);           →

pPtr++;
printf("%f\n", pPtr);           →
printf("%f\n", *pPtr);          →

pPtr += 2;
printf("%f\n", pPtr);           →
printf("%f\n", *pPtr);          →

qPtr = array + 4;
printf("%f\n", qPtr);           →
printf("%f\n", *qPtr);          →
```

```
\     qPtr-pPtr);     →
```

15. (10 points) The output of the code below is:

p == pe
pe != p

How could p be equal to pe and pe not equal to p?!!!!

a) Explain and state what the correct result should be.
b) Provide the corrective code.

```
class Person
{
	String name;
	Person(String name) {
		this.name = name;
	}
	public boolean equals(Object obj) {
		if (obj instanceof Person) {
			Person p = (Person)obj;
			return name.equals(p.name);
		}
		else {
			return false;
		}
	}
}
class PersonExtended extends Person
{
	String address;
	PersonExtended(String name, String address) {
		super(name);
		this.address = address;
	}
	public boolean equals(Object obj) {
		if (obj instanceof PersonExtended) {
			PersonExtended pe = (PersonExtended)obj;
			return name.equals(pe.name) && address.equals(pe.address);
		}
		else {
			return false;
		}
	}
}

public class MyClass {
	static public void main (String[] args) {
		Person p = new Person("Bob");
		PersonExtended pe = new PersonExtended("Bob", "Beirut");

		if (p.equals(pe)) {
			System.out.println("p == pe");
		} else {
			System.out.println("p != pe");
		}

		if (pe.equals(p)) {
			System.out.println("pe == p");
		} else {
			System.out.println("pe != p");
```

```
                    }
            }
    }
```