CSE 250: Data Structures
Fall 2015
Sample Final Exam

DIRECTIONS:

• Closed Notes. Closed Book. No electronics.
• Time Limit: 3 hours
• Make sure you write your NAME and UBIT above.
• Note that your UBIT is not your person number.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>/0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>/0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>/0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>/0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>/0</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>/0</td>
<td></td>
</tr>
</tbody>
</table>
1. **Short Answer** (0 x 0 = 0 points)

   (a) Give the asymptotic runtime of finding a particular element in an unsorted `std::vector`.

   (b) What algorithm can be used to find a particular element in a sorted `std::vector` in $O(\log(n))$ time?

   (c) Give the asymptotic runtime for insert, find, and delete for a `std::set`.

   (d) What is the asymptotic runtime of inserting into a `std::priority_queue`?

   (e) Draw the Binary tree given by the representations below.
   
   preorder: 6 3 2 1 5 4 13 19 15
   inorder: 1 2 3 4 5 6 13 15 19

   Is this a BST?
(f) Draw two different binary trees with the following level order traversal:
   10 11 15 9 21 53 1 2 3

(g) What is the expected asymptotic runtime to insert into a well designed hash table?

(h) Given a hash function that returns the same hash regardless of input, what happens to the insertion time into a the hash table?
2. **Scenarios** (0 × 0 = 0 points)

   (a) (0 points) Say you are writing a program that will store all customer’s current balance for a store. What backend data structure would you use to ensure the fastest lookup of a customer’s current balance when given their name? For implementation purposes you can think of their names as strings and their balances as doubles.

   (b) (0 points) Say you are writing a program to accept patients coming in to an emergency room. There’s only one doctor on staff and you need to ensure that all patients are seen with respect to their condition. For example even if someone has been waiting in line for a long time, if their condition is minor then those with more critical injuries will be seen first. What data structure would you use to efficiently implement this functionality?
3. **Code Reading and Runtime** (0 × 0 = 0 points)

(a) (0 points)

```c
void functionA(int n) {
    for (int i=0; i < n; i++){
        for (int j=0; j < n; j++){
            for (int k=0; k < n; k++){
                operation();
            }
        }
    }
}
```

i. (0 points) How many times is operation() called after executing functionA(1)?

ii. (0 points) How many times is operation() called after executing functionA(5)?

iii. (0 points) What is the asymptotic runtime of functionA(n)?
(b) (0 points)

```c
int functionB(int n) {
    operation();
    if (n > 0) {
        return functionB(n-1) + functionB(n-1);
    } else {
        return 1;
    }
}
```

i. (0 points) How many times is operation() called after executing functionB(1)?

ii. (0 points) How many times is operation() called after executing functionB(5)?

iii. (0 points) What is the asymptotic runtime of functionB(n)?
(c) (0 points)

```cpp
void functionC(int n) {
    std::set<int> finalSet;
    for (int i = 0; i < n; i++) {
        finalSet.insert(i);
    }
}
```

i. (0 points) What is the asymptotic runtime of functionC(n)?
4. Fix My Code (0 × 0 = 0 points)

Each of the following functions perform a specific task that is described in the comments preceding each definition along with an example input/output, but the functions contain errors.

Your goal is to fix each function by replacing only 1 line of code.

(a) (0 points)

```cpp
/*** Given an array of integers, find the product of its elements. */
Example: Given [2, 3, 1, 10] the output should be 60 ***/

```cpp
t int arrayElementsProduct(std::vector<int> inputArray) {
    int result = inputArray[0];
    for (int i = 0; i < inputArray.size(); i++) {
        result *= inputArray[i];
    }
    return result;
}
```

i. (0 points) What is the actual output for the given example?

ii. (0 points) Give the line number of the code that will be replaced.

iii. (0 points) Give the replacement code for this line that will fix the function.
(b) (0 points)

```cpp
std::vector<int> extractEachKth(std::vector<int> inputArray, int k) {
    std::vector<int> result;
    for (int i = 0; i < inputArray.size(); i++) {
        if (i == k) {
            result.push_back(inputArray[i]);
        }
    }
    return result;
}
```

i. (0 points) What is the actual output for the given example?

ii. (0 points) Give the line number of the code that will be replaced.

iii. (0 points) Give the replacement code for this line that will fix the function.
Given a map of int to int and an int k representing a value
returns a set of all keys that store the value k.

Example: inputMap = {(2,6), (12,5), (17,-4), (16,5)} and k=5
outputs the set {12,16} ***/

```cpp
std::set<int> findKeys(std::map<int, int> inputMap, int v) {
    std::set<int> result;
    for (std::map<int, int>::iterator it = inputMap.begin();
         it != inputMap.end(); it++){
        if ((*it).second == v){
            result.insert((*it).second);
        }
    }
    return result;
}
```

i. (0 points) What is the actual output for the given example?

ii. (0 points) Give the line number of the code that will be replaced.

iii. (0 points) Give the replacement code for this line that will fix the function.
5. Using Data Structures (0 × 0 = 0 points)

Write everything that will be displayed on the console after the following code runs. You can assume all the necessary include statements and using namespace std; have been added.

(a) (0 points)

```cpp
void printStack(stack<string> words){
    cout << endl << "Stack: " << endl;
    while(!words.empty()){
        cout << words.top() << endl;
        words.pop();
    }
    cout << endl;
}

void run()
{
    stack<string> sampleStack;
    sampleStack.push("Hello");
    sampleStack.push("Apple");
    sampleStack.push("Corvette");
    sampleStack.push("CSE250");
    sampleStack.push("Firetruck");
    printStack(sampleStack);
    sampleStack.push("Orange");
    sampleStack.push("Grape");
    sampleStack.pop();
    printStack(sampleStack);
}

int main()
{
    run();
    return 0;
}
```
(b) (0 points)

```cpp
void printStack(stack<string>& words)
{
    cout << endl << "Stack: " << endl;
    while(!words.empty()){
        cout << words.top() << endl;
        words.pop();
    }
    cout << endl;
}

void run()
{
    stack<string> sampleStack;
    sampleStack.push("Hello");
    sampleStack.push("Apple");
    sampleStack.push("Corvette");
    sampleStack.push("CSE250");
    sampleStack.push("Firetruck");
    printStack(sampleStack);
    sampleStack.push("Orange");
    sampleStack.push("Grape");
    sampleStack.pop();
    printStack(sampleStack);
}

int main()
{
    run();
    return 0;
}
```
(c) (0 points)

```cpp
void printQueue(queue<string>* words) {
    cout << endl << "Queue: " << endl;
    while (!words->empty()) {
        cout << words->front() << endl;
        (*words).pop();
    }
    cout << endl;
}

void run() {
    queue<string> sampleQueue;
    sampleQueue.push("Hello");
    sampleQueue.push("Apple");
    sampleQueue.push("Corvette");
    sampleQueue.push("CSE250");
    sampleQueue.push("Firetruck");
    printQueue(&sampleQueue);
    sampleQueue.push("Orange");
    sampleQueue.push("Grape");
    sampleQueue.pop();
    printQueue(&sampleQueue);
}

int main() {
    run();
    return 0;
}
```
class specialNumComparator {
private:
    int numFactors(int x) {
        int ret = 0;
        for(int i = 1; i < x; i++){
            if(x % i == 0) ret++;
        }
        return ret;
    }
public:
    bool operator()(int a, int b) {
        return numFactors(a) < numFactors(b);
    }
};

int main() {
    priority_queue<int, vector<int>, specialNumComparator> int_pq;
    int_pq.push(11);
    int_pq.push(24);
    int_pq.push(6);
    int_pq.push(18);
    int_pq.push(4);
    while(!int_pq.empty()) {
        cout << int_pq.top() << endl;
        int_pq.pop();
    }
    return 0;
}
class CustomComparator {
    public:
        bool operator()(const int &int1, const int &int2) const {
            return abs(int1 - 10) > abs(int2 - 10);
        }
};

void printPriorityQueue(priority_queue<int, vector<int>, CustomComparator> numbers) {
    cout << endl << "Priority Queue: " << endl;
    while (!numbers.empty()) {
        cout << numbers.top() << endl;
        numbers.pop();
    }
    cout << endl;
}

void run() {
    priority_queue<int, vector<int>, CustomComparator> samplePriorityQueue;
    samplePriorityQueue.push(12);
    samplePriorityQueue.push(25);
    for (int i = 0; i < 5; i++) {
        samplePriorityQueue.push(i);
    }
    samplePriorityQueue.push(9);
    printPriorityQueue(samplePriorityQueue);
    samplePriorityQueue.pop();
    samplePriorityQueue.push(14);
    samplePriorityQueue.pop();
    samplePriorityQueue.push(5);
    printPriorityQueue(samplePriorityQueue);
}

int main() {
    run();
    return 0;
}