1. This exam is closed book and no calculating devices of any type will be allowed. You are allowed, however, to prepare in advance the back of the hand-out info page with whatever notes you wish to place on it, and you may use this page during the exam. When the exam is over, submit this sheet with the rest of your exam.

2. Print your full name and your email address in the boxes above.

3. Print your name at the top of every page.

4. Please write clearly and legibly. If I can’t read your answer, I can’t give you credit.

5. Remember, these are NOT necessarily in order of difficulty. Please read all the problems first, and don’t allow yourself to get stuck on a single problem.

<table>
<thead>
<tr>
<th>#</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Score</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
</tbody>
</table>
1. (a) (10 points) Fill in the diagram below to represent the underlying memory configuration that is present after the following commands are executed (Note: identifiers should be represented by \{a, b, c, d, e\} not containing pointer(\*) or reference(&) symbols):

```c
char a(‘X’); 
char b(a); 
char *c(&b); 
char &d(*c); 
char *e = new char(‘Y’); 
```

<table>
<thead>
<tr>
<th>Identifiers</th>
<th>Value</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>\text{X}</td>
<td>\text{241}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{242}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{243}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{244}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{245}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{246}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{247}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{248}</td>
</tr>
</tbody>
</table>

(b) (10 points) Based upon your solution to the above problem, fill in the below diagram to portray the updated configuration after the completion of the following additional commands.

```c
a = ‘L’; 
c = e; 
d = ‘P’; 
```

<table>
<thead>
<tr>
<th>Identifiers</th>
<th>Value</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>\text{241}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{242}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{243}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{244}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{245}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{246}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{247}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\text{248}</td>
</tr>
</tbody>
</table>
2. (20 points) What output is generated by the following code fragment when main is executed:

```cpp
#include <iostream>
using namespace std;

void f(int x);
void g(int& y);

void f(int x) {
    cout << "x is " << x << endl;
    x = x+1;
    cout << "x is " << x << endl;
    g(x);
    cout << "x is " << x << endl;
}

void g(int& y) {
    cout << "y is " << y << endl;
    y = y+2;
    cout << "y is " << y << endl;
}

int main() {
    int z = 5;
    cout << "z is " << z << endl;
    f(z);
    g(z);
    cout << "z is " << z << endl;
}
```

Output:

```
2
```
3. (20 points) Suppose we have an initially empty stack $S$ and an initially empty queue $Q$. Fill in the following table for the following sequence of commands.

<table>
<thead>
<tr>
<th>Operation</th>
<th>$S$ bottom $\rightarrow$ top</th>
<th>$Q$ front $\leftarrow$ rear</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q.push($A$)</td>
<td>( )</td>
<td>(A)</td>
</tr>
<tr>
<td>Q.push($B$)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q.push($C$)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.push(Q.front())</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q.pop()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.push(Q.front())</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q.pop()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q.push(S.top())</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.pop()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.push(Q.front())</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q.pop()</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. (20 points) Complete a program to reverse a queue using a stack. Use STD queue and stack and operations \texttt{push}, \texttt{pop}, \texttt{empty}, \texttt{front(queue)}, \texttt{top(stack)}. Finally print queue elements in reverse order as below:

```
#include<iostream>
#include<stack> // std::stack
#include<queue> // std::queue
using namespace std;

int main() {
    // define queue and stack ADT
    queue <int> Q;
    stack <int> S;

    // enqueue Q
    Q.push(1);
    Q.push(3);
    Q.push(6);
    Q.push(9);
    Q.push(11);

    // finally print queue elements in reverse order
    while (!Q.empty()) {
        cout << Q.front() << " ";
        Q.pop();
    }
    cout << endl;
    return 0;
}
```
5. (a) (10 points) Give an analysis of the running time (Big-Oh notation) for each of the following 3 program fragments
   
   i. sum = 0;
   for(i=0;i<n;i++)
      sum++;
   for(j=0;j<n;j++)
      sum++;

   ii. sum = 0;
    for (i=0;i<100;i++)
       sum++;

   iii. sum = 0;
    for(i=0;i<n/2;i++)
       for(j=i;j<n/4;j++)
          sum++;

   (b) (10 points) If it takes 1 second to run program iii for n=50, how long will it take to run for n=500 (seconds)?
(scratch paper)
(scratch paper)