FSU COP 4530 / CGS 5425 (Fall 14)
Data Structures, Algorithms, and Generic Programming

Midterm: Max points: 100, Time: 45 minutes

First Name: _______________ Last Name: ________________

This is a closed book examination.

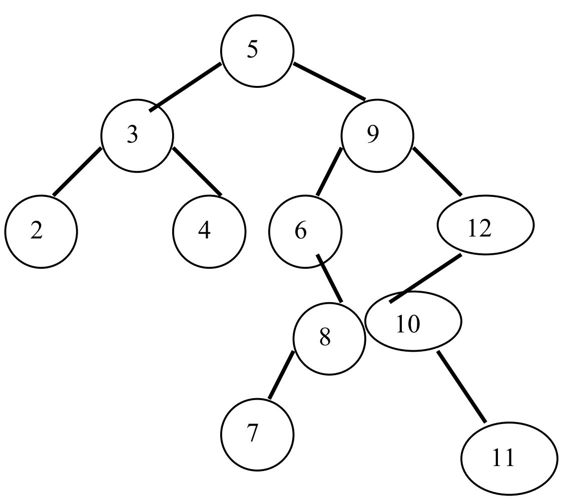
1. (a) (10 points) Let us perform the following operations on a stack: `push(1)`, `push(2)`, `push(3)`, `pop()`, `pop()`, `push(4)`, `push(5)`. (i) Draw a figure to show the state of the stack after these operations have completed. (ii) If the underlying container is a circular array with `push_back` on the array used to implement `push` on the stack, then which circular array operation should be used to implement `pop` on the stack.

(i)

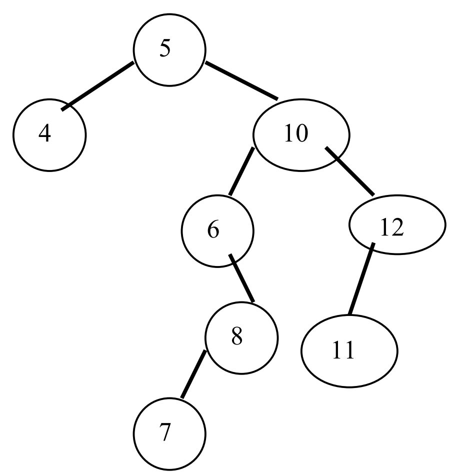
5
4
1

(ii) pop_back

(b) (20 points) Draw the Binary Search Tree that results after the following sequence of operations on a tree that is initially empty, using algorithms discussed in class: *insert(5), insert(9), insert(12), insert(10), insert(3), insert(4), insert(2), insert(6), insert(11), insert(8), insert(7), Delete(2), Delete(3), Delete(9)*.



After inserts



Final

2. (a) (10 points) Let the time complexity of an algorithm be $n^3 \log^2 n + 200n + 20\,n^4 + 5000$. Give a good asymptotic time complexity in big-O notation.

$O(n^4)$

(b) (10 points) Derive the time complexity for computing f(n) for the following recursive function. Show all steps.

```
f(1) = 1
f(n) = n² + 3 f(n-1),  n > 1
```

```
t(1) = 1
t(n) = 1 + t(n-1), n > 1
t(n) = 1 + t(n-1) = 2 + t(n-2) = … = k + t(n-k).
```

Take k = n-1. This gives:

```
T(n) = n-1 + t(1) = n = O(n)
```

(c) (10 points) Give good asymptotic time complexities for each of the following operations on the data structures given below.

Av.: Average, Amort.: amortized, WC: Worst case, x: Ignore

	push	*Push front*	*Push back*	*Pop back*	*search*
Sorted vector	Amort: n	x	x	x	log n
vector	x	Amort: n	Amort: 1	1	n
stack	1	x	x	x	x
BST	WC: n Av: log n	x	x	x	WC: n Av: log n
doubly linked list	x	1	1	1	n

3. a. (5 points) Which header file will you include in order to use the STL linked list?

list

b. (5 points) Declare a variable which is a vector of ints.

vector<int> V;

c. (5 points) Write a statement that places 5 into the end of the array in the object you declared above.

V.push_back(5);

d. (5 points) Declare an iterator for a vector of ints and write code that uses it in a loop to output all the elements in the vector. You should not use the vector's bracket operator.

```
for(vector<int>::iterator I = V.begin(); I != V.end(); ++i)
        cout << *I << endl;
```

4. (20 points) Let DLL be a self-organizing doubly linked list class *without sentinels*, using a PreviousFront method. In the PreviousFront method, when we search for a node and find it, we will move its *previous* node to the front of the list, if a previous node exists. Otherwise, the list is unchanged. Implement a member function called PreviousFront, which is given as argument a pointer to a node that has been found through a search. This function implements the operation that performs the self-organization. You may assume that the linked list stores only ints; consequently, your code need not be templated. You can also assume reasonable fields in the DLL class and in its Node class, such as Node *Head, Node *Tail, Node *next, and Node * previous respectively.

```
void PreviousFront(Node *N){

  if(Head == N || Head == N->prev )
   return;
  Node *M = N->prev;

  M->prev->next = N;
  N->prev = M->prev;
  M->next = Head;
  Head->prev = M;
  Head = M;
  M->prev = 0; // 0 or NULL are ok
}
```