CMPT 117 (02)
Instructor: Kevin Grant

Midterm Examination
February 22, 2006

Name: Craig Bloch-Hansen

Student Number: 147742

Rules:

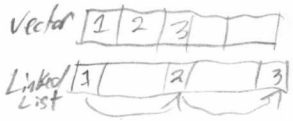
1) You have 50 minutes to complete the exam. You have approximately one minute per mark, so allocate your time wisely
2) This is a closed book examination. You may not use any reference material (electronic or non-electronic) of any kind.
3) You may not communicate with others
4) Cheating is not tolerated, and will result in a forfeiture of the exam, and a 0% grade.

Good luck!!

Part I: Theory (10 Marks)

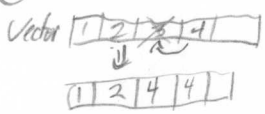
1. (4 Marks) In class, we discussed four advantages that linked lists have over vectors. Name two of them, and explain with a small example/illustration.

(1) Do not need large blocks of contigous memory in heap,
- Vectors are arrays and need to have sequential memory where as Linked lists ~~have~~ pointers to the next part of the list. Each ~~item~~ in the list is stored as a node made up of data and a pointer to the next node in the list.

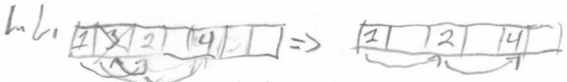


4

(2) - Do not need to copy and shift information when deleting,



- need to copy 4 into 3's position; this is a linear operation



- deletes node and pointer and changes pointer of previous node.
Constant time operation

2. (3 marks) When a class uses heap memory, what three methods must we always include in order to ensure correct functionality?

0

3. (3 Marks) Compare and contrast our *container* class with our *vector* class, using our following table:

	Insertion	Deletion	Search
Container	O(c)	O(c)	O(n)
Vector	O(n)	O(n)	O(n)

3

Part II: Code Tracing (5 Marks)

Often in debugging, many programmers (who do not have access to Visual Studio) will leave little *cout* commands to help them find errors in code. In the following code segments, *cout* commands have been left in order to follow what the code is doing.

a) (3 marks) What would the output of this code be?

```
int f1(int x) {
   cout << "Calling f1(int)" << endl;
   return x * x;
}

double f1(double x) {
   cout << "Calling f1(double)" << endl;
   return x * x;
}

float f1(float x)  {
   cout << "Calling f1(float)" << endl;
   return x * x;
}

int main() {

   f1(2.0f);
   f1(2);
   f1(2.0);

   f1(f1(2.0f) * f1(2.0));

}
```

Output:

Calling f1(float)
Calling f1(int)
Calling f1(double)
Calling f1(float)
Calling f1(double)
Calling f1(double)

3

b) (2 marks) If there were no cout commands in our f1 functions, write a function template for f1 that would take the place of all three functions above.

```
template<Item> typename
Item f1(Item x)
{return x*x;}
```

1.5