```
Christopher Wilcox
Assignment 3
CS368 Spring 2010
Due: 2010/03/13
```

1. For each of the following declaration and initialization statements, identify what is declared as constant: the pointer variable's value, or what the pointer points to.

```
        const int * pX = &X;
            pX points to a constant integer X
         int * const pX = &X;
            constant pointer pX to integer X
         int const * pX = &X;
            pointer to a constant integer X
```

2. Which section (segment) of memory is a C++ programmer expected to manage?

        A C++ programmer is expected to manage the heap.

   Give a brief definition of a memory leak.

        A memory leak occurs when space is alloated and not deallocated even after losing
        pointers to that area of memory.

   Why are memory leaks bad?

        Memory leaks are bad because they 'bloat' a program as it runs. This is bad as
        it is a poor utilization of memory.

3. What happens when the following function is executed?

```
    void fcn() {
      int *p;
      int a = 5;

      while (1) {
        p = new int;
        p = &a;
        a++;
      }
    }
```

        Memory is leaked as there is not a delete operation for every new operation. The
        programmer should better manage memory.

4. Briefly, why do we need the scope resolution operator (::)?

        The scope resolution operator is used to define already declared member functions.
        This allows us to define the normal functions of the class

5. Under what circumstances is the default destructor insufficient at preventing memory leaks?

        The default destructor does not follow pointers. This can cause memory leaks. If
        this is the case, you should create a destructor to do properly handle the case.

6. Show how to modify this code such that data encapsulation is preserved.

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

class myClass {
  private:
      int a;
  public:
    void setval(int u) { a = u; }
    void updateval(int u) { a = a + u; }
    void printval() {
      cout<<"\nThe updated value is : " << a << endl;
    }
};

int main(void) {
  myClass obj1;

  obj1.setval(100);
  obj1.updateval(2);
  obj1.printval();
  return 0;
}
```

    Encapsulationis the process of combining data to a class. This allows us to hide
    certain details, and provide a simple way to update data. This also allows us to hide
    implementation. To accomplish this, the variable myClass.a is made private and a setval
    method is provided to modify the value. Above is the modified code.