Problem 1 (10 points): Please answer the following true/false and short answer questions.

{(T) F} C++ provides a default copy constructor.

{T (F)} Default values for the arguments of a function can only be used with global functions. You cannot use default arguments with constructors and member functions.

{(T) F} When you make a class in C++, you are automatically provided by the compiler with a working overload of the assignment operator.

{T (F)} Friend functions are used to break encapsulation, but they still cannot access private data directly.

{T (F)} The new operator returns a *name* handler to the data that is dynamically allocated.

{Fill in the blank}
Using the keyword 'this' inside a member function stores the value of the pointer to the object that is used to call the member function.

{Short Answer}
Why would you ever pass a variable by reference AND designate it as constant, which is roughly illustrated in the following global function.

```
void foo (const barType & bar)
{ bar.memberfunction1();
  bar.memberfunction2(); }
```

It is passed by reference for performance reasons, but it is designated a constant so that it cannot be changed inside the function.

{Short Answer} What is the rule of three?

It states that...

If you need to provide a custom destructor, assignment operator, or copy constructor, then you typically must provide all THREE!

Problem 2 (15 points): Please add code where indicated in the comments.

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

int main()
{
double * dynamicArray;
int size;

cout << "Please input the size of the array of doubles: " ;
cin >> size;

//provide code here to dynamically allocate this array
//make sure that dynamicArray points to the beginning of the array.
```

```
dynamicArray = new double[size];
```

```
//Provide code here to fill in the array with values of -100.0
```

```
for(int i=0; i < size; i++)
    dynamicArray[i] = -100.0;
```

```
//Provide code to print out the entire array. Have
//each value print to the terminal separated by spaces
```

```
for (int i=0; i<size; i++)
    cout << dynamicArray[i] << " ";
cout << endl;
```

```
//Anything else you need to do before ending the program? If so, please
//add it here.
```

```
delete [] dynamicArray;
```

```
return 0;
}//end main
```

Problem 3 (15 points): Consider the following short program that is not written in an object oriented programming style. On the next page I would like for you to re-write this program using a object oriented programming style. I would like for you to have one class that you create called Person. Please have a constructor and a member function that prints out the information. You only have to create set and get functions that you need. Also you can assume your solution is all in one file, and it must have the same functionality as program below.

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

int main()
{
int agePerson1 = 42;
int agePerson2 = 21;
double heightInchPerson1 = 70;
double heightInchPerson2 = 71;
string namePerson1 = "Malik";
string namePerson2 = "Justin";

cout << namePerson1 << "  " << agePerson1 << " years " << heightInchPerson1 << " inches" << endl;
cout << namePerson2 << "  " << agePerson2 << " years " << heightInchPerson2 << " inches" << endl;
cout << "The average age of these two people is : " << 0.5*(agePerson1+agePerson2)<< endl;
return 0;
}
```

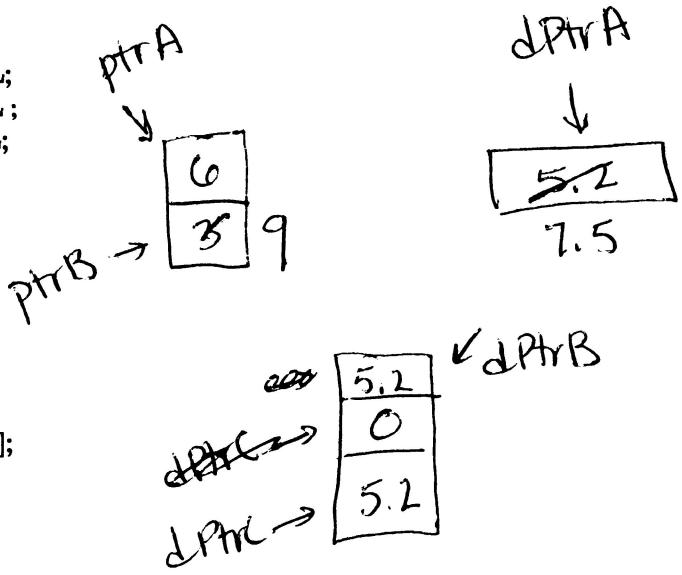
```
#include <iostream>
#include <string>
using namespace std;

Class Person
{
    public:
    Person (int a, int h, string n): age(a), height(h), name(n) {}

    Void print()
    { cout << name << " " << age << " years" << height << "inches" << endl; }
    private:
     int age;
     int height;
     string name;
};

int main()
{   Person person1 (42, 70, "Malik");
    Person  person2 (21, 71, "Justin");
    person1.print();
    person2.print();
    cout << "The average age of these two people is: ";
   Cout << 0.5 * (person1.getAge() + person2.getAge()) << endl;
```

Problem 4 (15 points): Pointermania!! Please show the output of the following program in the space provided below. Also do you think there are any memory leaks?

9 5.2 7.5 9 No memory leaks!

0 → or garbage

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

int main()
{
int * ptrA = NULL;
int *ptrB = NULL;
double numA = 5.2;
double *dPtrA = NULL;
double *dPtrB = NULL;
double *dPtrC = NULL;

ptrA = new int[2];
ptrA[0] = 6;
ptrA[1] = 3;
ptrB = ptrA+1;


dPtrA = &numA;
dPtrB = new double[3];

*dPtrB = *dPtrA;
dPtrC = dPtrB;
dPtrC++;

*(++dPtrC) = *dPtrA;
*(&numA) += 2.3;
*ptrB = *dPtrB + 4;

cout << *ptrB << ' ';
cout << *dPtrB << ' ';
cout << *dPtrA << ' ';
cout << *(ptrA+1) << endl;
cout << *(--dPtrC) << endl;

delete [] ptrA;
delete [] dPtrB;

return 0;
}
```



Problem 5 (5 points): Consider the following class interface:

```
class Doggie
{
public:
  void walkDog(int);
private:
   int numberOfSniffs;
} ;
```

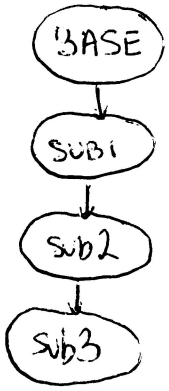
Also, consider the following main function implementation:

```
int main()
{
int minutesWalked;
Doggie * DoggiePtr = new (Doggie);
//more code here not included
}
```

If in main(), you would like to call the member function courseSurvey() in class Ece2036, which of the following statements could you use?

a. DoggiePtr.walkDog(minutesWalked);
b. DoggiePtr.->walkDog(minutesWalked);
c. DoggiePtr->walkDog(minutesWalked);
d. (*DoggiePtr). walkDog(minutesWalked);
e. c&d
f. b&d
g. none of the above

Problem 6 (15 points): Consider the following code. Please show output to on the right side of this page.




```
#include <iostream>
using namespace std;
class Base
{public:
  virtual void Func1() = 0;
  void Func2();
  virtual void Func3();
  void Func4(); };
class Sub1 : public Base
{public:
    virtual void Func1();
    void Func2(); };
class Sub2 : public Sub1 {
public:
  virtual void Func1();
  void Func2();
  virtual void Func3(); };
class Sub3 : public Sub2
{public:
  virtual void Func1();
  void Func2(); };
void Base::Func2()
{ std::cout << "Hello from Base::Func2()" << std::endl; };
void Base::Func3()
{ std::cout << "Hello from Base::Func3()" << std::endl; };
void Base::Func4()
{ std::cout << "Hello from Base::Func4()" << std::endl; };
void Sub1::Func1()
{ std::cout << "Hello from Sub1::Func1()" << std::endl; }
void Sub1::Func2()
{ std::cout << "Hello from Sub1::Func2()" << std::endl; };
void Sub2::Func1()
{ std::cout << "Hello from Sub2::Func1()" << std::endl; };
void Sub2::Func2()
{ std::cout << "Hello from Sub2::Func2()" << std::endl; };
void Sub2::Func3()
{ std::cout << "Hello from Sub2::Func3()" << std::endl; };
void Sub3::Func1()
{ std::cout << "Hello from Sub3::Func1()" << std::endl; };
void Sub3::Func2()
{ std::cout << "Hello from Sub3::Func2()" << std::endl; };

int main()
{
 Sub2 objectSub2;
 Sub3 objectSub3;
 Base * ObscuringObject1 = &objectSub2;
 Base & ObscuringObject2 = objectSub3;

 ObscuringObject1->Func1();
 ObscuringObject1->Func2();
 ObscuringObject1->Func3();
 ObscuringObject1->Func4();

cout << endl;

 ObscuringObject2.Func1();
 ObscuringObject2.Func2();
 ObscuringObject2.Func3();
 ObscuringObject2.Func4();
}
```

derived is objectSub2

derived is objectSub3

Hello from Sub2 :: Func1()
Hello from Base :: Func2()
Hello from Sub2 :: Func3()
Hello from Base :: Func4()

Hello from Sub3 :: Func1()
Hello from Base :: Func2()
Hello from Sub2 :: Func3()
Hello from Base :: Func4()

Problem 7 (15 points): Please trace through this code and clearly show the output of this program in the box provided to the right of the code.

```
#include <iostream>
using namespace std;
class A{
public:
  A();
  A(int);
  ~A();
  A operator+(const A& rhs) const;
  void print();
private:
 int x;
};
A::A(): x(0){ cout<<"A Created "<<endl; };
A::A(int x): x(x){ cout<<"A Created with "<< x <<endl; };
A::~A(){ cout<<"A Destroyed with "<< x <<endl; };
void A::print() { cout << x  << " ";}
A A::operator+(const A& rhs) const
{ return A(x+rhs.x); }

class B{
 public:
  B();
  B(int);
  ~B();
  B operator+(const B  &rhs) const;
  void print();
 private:
  int x;
};
B::B(): x(0){ cout<< "B Created"<<endl;};
B::B(int x): x(x){ cout<< "B Created with "<< x <<endl;};
B::~B(){ cout<< "B Destroyed with "<< x <<endl;};
void B::print() { cout << x << " ";}
B B::operator+(const B & rhs) const
{ return B(x + rhs.x); }

int main()
{
 A a(15);
 B c(1);
 B d(2);
 A b(a);

 a.print();
 b.print();
 a = b + a;
 c.print();
 d.print();
 d = d + c;
 cout << endl;
}
```

a	c	d	b
~~15~~ 30	1	~~2~~ 3	15

a --> A created with 15
c --> B created with 1
d --> B created with 2
15 15 A created with 30
A destroyed with 30
1 2 B created with 3
B destroyed with 3

space →

A destroyed with 15
B destroyed with 3
B destroyed with 1
A destroyed with 30

Problem 8 (10 points): For the code in problem 7, please show the implementation for a copy constructor for class A. When this copy constructor is used I would like for you to have it work in an unusual way. I don't want it to make an exact copy, in fact I would like for it to add 10 to the private data member that it is copying. For example, if a class A object (call it object1) is instantiated and has a value of 5, then the declaration of

```
A object2(object1);
```

would put the value of 15 into the object2. Show the implementation of the copy constructor below.

```
A::A(const A &objectToCopy)
{
    x = objectToCopy.x + 10;
}
```