CSci 1113
Final

Name: ________________________________

Student ID: __________________________

Lab Section (circle one): W5:45, Th8:00, Th11:15, Th2:30, Th5:45

Instructions: Please pick and answer any 10 of the 12 problems for a total of 100 points. If you answer more than 10 problems, only the first 10 will be graded. The time limit is 120 minutes. Please write your answers in the space provided. The exam is open book and notes. You may use electronic devices to ONLY look at either an e-book version or electronic notes. You may not use the internet, compiler or any other outside resources. (If you are typing on your keyboard/input device for anything other than ctrl-F to find words in the e-book or notes, this is probably not acceptable.)
**Problem (1)** [10 points] Write (in C++) a getSlope() member function for the class below (i.e. the function should be inside the class). A slope can be found by this formula: \( \frac{y_2 - y_1}{x_2 - x_1} \). Simply write the definition of getSlope(), you do not need to declare it.

```cpp
class Problem1 {
private:
    double xStart;
    double xEnd;
    double yStart;
    double yEnd;
public:
    Problem1();
};

double Problem1::getSlope()
{
    return (yEnd-yStart)/(xEnd-xStart);
}
```

1 pt: return type
3 pt: properly in the class (Problem1::)
1 pt: function name/arguments correct
2 pt: returning the same type as function return type
3 pt: formula used properly in return
Problem (2) [10 points] Write (in C++) a useful non-default constructor for the following class (just the actual function definition, you do not need to declare it):

```cpp
class Problem2 {
private:
    char symbol;
    string name;
    double speed;
public:
    Problem2();
};
```

```cpp
Problem2::Problem2(char s, string n, double sp) {
    symbol = s;
    name = n;
    speed = sp;
}
```

2 pt: properly in class (Problem2::)
2 pt: function name (and no return type)
2 pt each: properly inputting and setting each member variable (symbol, name, speed)
Problem (3) [10 points] Suppose a class “Problem3” is defined as below, along with the code segment inside the default constructor. Write an appropriate deconstructor in C++.

class Problem3 {
private:
    int* x;
public:
    Problem3();
};

// somewhere inside the constructor
    x = new int[10];

Problem3::~Problem3()
{
    delete [] x;
}

2 pt: properly in class (Problem3::)
3 pt: function name and arguments correct (and no return type)
3 pt: delete on x
2 pt: using [] in delete
Problem (4) [10 points] What is the output of the following segment of code. What part (if any) should you use `delete` on from this segment? Explain your reasoning.

```cpp
int a = 5;
int b = 2;

int* x = &a;
int* y = &b;

y = x;
*y = *x * *y;

cout << a << " " << b << endl;
cout << *x << " " << *y << endl;
```

25 2
25 25

There should be no deletes as the pointers reference non-dynamic (static?) variables. You only use `delete` on `new`s.

6 pt: correct cout answers (1.5 per answer, partial credit for work shown)
4 pt: reasoning why not to delete
Problem (5) [10 points] Write (in C++) an overload function for the “>>” operator (i.e. for cin) for the following “Point” class to give values to all variables inside the class (i.e. member variables). This time you do need to declare the function in the class (you can draw an arrow from some text into the class given below) along with the definition.

```cpp
class Point {
private:
    double x;
    double y;
public:
    Point();
    void setX(double newx);
    void setY(double newy);
    double getX(); // returns x value
    double getY(); // returns y value
};

friend istream& operator>>(istream& in, Point& p) // in class declaration
{
    in >> p.x >> p.y;
    return in;
}
```

2 pt: declaration has friend
2 pt: function name
2 pt: arguments correct (and in correct order) (-1 if missing & for istream)
1 pt: return type correct
2 pt: reading x and y from in
1 pt: returning in correctly
Problem (6) [10 points] Write (in C++) a midPoint() function that takes as input two Point variables (as defined in problem 5). This function should correctly find and return the midpoint of the two Points passed in as input (the formula for this case is: \( x_{\text{midpoint}} = \frac{x_1 + x_2}{2}, y_{\text{midpoint}} = \frac{y_1 + y_2}{2} \)). An example code of its use is given below:

```cpp
Point p1;
Point p2;
// magically initialize p1 and p2 to something
Point mid = midPoint(p1, p2);
```

```cpp
Point midPoint(Point p1, Point p2)
{
    Point result;
    result.setX((p1.getX() + p2.getX())/2);
    result.setY((p1.getY() + p2.getY())/2);
    return result;
}
```

2 pt: return type correct
3 pt: function name and argument types correct
1 pt: making a new Point to return
2 pt: setting values of new point correctly
2 pt: returning point
Problem (7) [10 points] Write (in C++) a evenArray() function, which returns a dynamically created array of even numbers (starting from 0 going up) of the size specified by the input. For example, for the code segment below the array length should be 10 (containing numbers 0 through 18).

```c++
int* evenNumbers = evenArray(10);
```

```c++
int* evenArray(int size)
{
    int* result = new int[size];
    for(int i=0; i < size; i++)
    {
        result[i] = i*2;
    }
    return result;
}
```

1 pt: int* return type for function
1 pt: function name
1 pt: function arguments
3 pt: using new to dynamically create an array
1 pt: dynamic array correct size
2 pt: array has correct values (my for loop)
1 pt: returning array
Problem (8) [10 points] Find 3 possible places for errors in the following code. Assume this is all the code except for namespaces and includes. Explain specifically what causes each error and whether it is a syntax or logic error:

class Problem8 {
private:
    double y;
    getY();
};

double getY()
{
    return y;
}

int main()
{
    Problem8 x;

    cout << getY();

    return 0;
}

g getY() is private // logic (or syntax based on how main() uses it)
double getY(); // syntax, in class definition
double Problem8::getY() // syntax, function definition
cout << x.getY(); // syntax, in main()

3 pt per problem found (2 for error, 1 for error type)
1 pt free
Problem (9) [10 points] Write (in C++) a recursive function eachDigit() that will cout each digit of the input integer on a separate line.

```cpp
int main()
{
    eachDigit(456);
    // above should cout (note it is reversed):
    // 6
    // 5
    // 4
}

void eachDigit(int number)
{
    cout << number%10;
    if(number < 10)
    {
        return;
    }
    eachDigit(number/10); // note: it is also acceptable if they put the cout after the recursive call
}
```

1 pt: function name
1 pt: int argument
2 pt: cout statement shows a single digit
2 pt: stopping case is correct
2 pt: recursive call
2 pt: argument in recursive call has one less digit
**Problem (10)** [10 points] Write a segment of code in C++ (i.e. pretend you are writing somewhere in main()) that creates the following 10 by 10 array. The first row and column (i.e. top row and left column) should be 1. Every other cell in the array should be the sum of the cell to the left plus the cell above itself.

```cpp
int grid[10][10]; // grid can also be a double
for(int i=0; i < 10; i++)
{
    grid[i][0] = 1;
    grid[0][i] = 1;
}

for(int i=1; i < 10; i++)
{
    for(int j=0; j < 10; j++)
    {
        grid[i][j] = grid[i-1][j] + grid[i][j-1];
    }
}
```

1 pt: grid is a valid type
2 pt: grid is 10 by 10;
2 pt: initialize first row/column (could actually initialize whole array to 1)
3 pt: double for loop
2 pt: filling in grid correctly
Problem (11) [10 points] Write (in C++) a findMax() function that takes in three integers as inputs. This function must have a *void* return type and set the third input integer to the maximum of the first two input integers. An example use is below:

```cpp
int a = 2;
int b = 10;
int x;

findMax(a, b, x);
cout << x << endl; // should show 10
```

```cpp
void findMax(int a, int b, int &x)
{
    if(a > b)
    {
        x=a;
    }
    else
    {
        x=b;
    }
}
```

1 pt: void return type (reading directions...)
1 pt: function name
3 pt: all 3 arguments are ints
2 pt: using & for x
1 pt: having an if to find the max
2 pt: setting x to max correctly
Problem (12) [10 points] Write (in C++) a single if-statement that is true on the set of integers shown. A “...” signifies that the pattern continues in the specified direction (i.e. a “...” on the left indicates the pattern continues for smaller numbers). An example is provided below.

int i: 0, 1, 2, 3, 4, ...

Answer:
if(i >= 0)
or
if(i > -1)

(a) int i: ... -10, 0, 10, 20, 30, 40, ...

(b) int i: ... -2, 2, 4, 6, 8, 12, 14, 16, 18, 22, ...
(look carefully)

(c) int i: 0, 1, 2, 3, 4, 6, 7, 8, 9, 10
(look carefully)

(d) int i: ... -8, -6, -4, -2, 12, 14, 16, 18, 20, ...
(you might want parenthesis for this)

(e) int i: 1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15
(parenthesis might be necessary here too)

(a) if(i%10==0)
(b) if(i%2==0 && i%10!=0)
(c) if(i >= 0 && i <= 10 && i != 5)
(d) if(i%2==0 && (i < 0 || i > 10))
(e) if( (i > 0 && i < 6) || (i > 9 && i < 16))

2 pt each: correct... (give partial credit for closeness)