

Name: _____

SSN: _____

CS 12 - Final December 10, 1997

Be sure to read each problem carefully and follow the directions. Points may be marked off if you do not follow the directions. For example, if the problem asks you to write a function, do not write an entire program. Please feel free to ask if you have any questions.

Problem 1	30	
Problem 2	6	
Problem 3	6	
Problem 4	6	
Problem 5	24	
Problem 6	8	
Problem 7	10	
Problem 8	10	
Problem 9	12	
Problem 10	10	
Problem 11	40	
Problem 12	28	
TOTAL	190	

1. (30 pts) True/False (please write out “True” and “False”, do not just put a T or F).

(a) When the following statements are executed,

```
int x;
cout << ‘Enter a value: ‘;
cin >> x;
cout << setfill(‘*’) << x << endl;
```

you will see an asterisk in the output.

(b) After using the extraction (>>) operator, the input stream buffer is empty.

(c) Anything that can be written recursively can be written iteratively.

(d) Inheritance is when a class contains another class as a member.

(e) The following two functions can be overloaded.

```
void func(int, double);
void func(double, int);
```

(f) The following two functions can be overloaded.

```
int func2(int x, char c){}
void func2(int y, char ch){}
```

(g) If one inherits a base class using protected mode, all members of the base class will be protected in the derived class.

(h) When inheritance has been used, the constructor of the base class is called first.

(i) The following statement will print the integer x in binary form and hexadecimal form.

```
printf(‘%x\t%b\n’, x);
```

(j) A string is a character array.

(k) *strncpy* will ensure that the destination string is NULL-terminated.

(l) *strcat* will ensure that the newly concatenated string is NULL-terminated.

(m) All constructors have a void return type.

(n) If class A is a friend of class B, then class B is a friend of class A.

(o) Given the following definition

```
class C
{
    ...
    friend D;
};
```

class C has access to D’s private members.

2. (6 pts) Fill in the blank.

- (a) When using inheritance, the new class is called the _____ class.
- (b) When a width is set, the default justification is _____ justified.
- (c) The default control access of a class member is _____.
- (d) A class object is typically passed by _____.
- (e) The choice of what function to call, when using a virtual function, is made at _____ time.
- (f) A template function is generated at _____ time.

3. (6 pts) List 3 uses of pointers.

4. (6 pts) Explain the difference between the following functions.

```
getline(char *buffer, int max, char delimiter);  
get(char *buffer, int max, char delimiter);
```

5. (24 pts) Define the following terms.

(a) Batch Program

(b) String

(c) Pointer

(d) Dynamically allocated array

(e) Constructor

(f) Public Access (class member definition)

(g) Public Inheritance

(h) This pointer

6. (8 pts) Write the long form of the following function calls (translate to use the entire function name). Use the following declarations.

```
class TempClass
{
    ...
};
```

```
int A, B, C;
TempClass C1, C2;
```

- (a) `cout << 'Hello';`
- (b) `A = B + C;`
- (c) `C1 = C2;`
- (d) `cout << C1;`

7. (10 pts) Show the values of each element in the array A after the following code has been performed.

```
int A[5] = {20, 2, 16, 1, 5};
int *ip;
int x = 2;

ip = A;
*ip = 3;
ip++;
*ip = *(ip + x);
ip++;
*ip = *ip - 1;
ip+=x;
*ip = 100;
```

8. (10 pts) Given the following code:

```
class Grade
{
private:
    char name[30];
    int mid1, md1max;
    int mid2, md2max;
    int final, finmax;
    char finalgrade;
public:
    Grade();
    void set_midterm1();
    void set_midterm2();
    void set_final();
    void calculate_final_grade();
};
```

```
Grade roster[30];
int index = 4;
Grade *gr = new Grade;
gr = &roster[index];
```

What are the types of the following expressions? Write in the letter of the corresponding correct answer. An answer may be used more than once.

- | | | |
|--------|--------------------|----------------------|
| ___ 1 | roster[index] | A int |
| ___ 2 | Grade | B character |
| ___ 3 | roster | C character array |
| ___ 4 | roster[index].mid1 | D string |
| ___ 5 | gr | E Grade |
| ___ 6 | gr.finalgrade | F Grade pointer |
| ___ 7 | roster.md1max | G address of a Grade |
| ___ 8 | roster[20].name | H array of integers |
| ___ 9 | gr->name[2] | I array of Grades |
| ___ 10 | final | J illegal |

9. (12 pts) Write a recursive function to return the number of digits in a positive integer. Ex. 11 has 2 digits, 123 has 3 digits, 9876 has 4 digits. (Hint, think about the mod and div operators.)

10. (10 pts) Write a template function to exchange the values of two objects.

11. (40 pts) You will be writing a program that reads in values from a file and stores them in an array. Then a function will be called to reverse the array. Lastly, the reversed array will be printed out.

Write a class that allows you to keep track of a dynamically allocated array of doubles. You should maintain the maximum size, the current size, the name of a file to read from, and the array. Your member functions should include the following.

- (a) A constructor should allow the user to set the size.
- (b) An appropriate destructor.
- (c) A member function called *Set_and_Read* should take a parameter for the name of the file to be read from, set the data member, open the file, and read in and store values into the array. The file may be in any format including many on one line and blank lines at the end. Do not overfill the array.
- (d) A member function called *Reverse* should reverse the array. The reversed array should be stored in the original array. Your function should not have any output.
- (e) Overload the insertion stream operator to print the array. Each double should appear on a separate line.

11 cont.

Now fill in the following short main program

```
void main()
{
    // Prompt for size of the array.

    // Create an object of your class
    // type of the specified size .

    // Prompt for the name of the file
    // to read values from.

    // Call function to set file name
    // and read values from the file.
    // Print the array.

    // Reverse the array.
    // Print the reversed array.
}
```


12. Be sure to follow all directions closely.

- (a) (10 pts) Write a class to represent an address. Your class should be called *Address* and should contain two data members, an integer to stand for the number and a string to stand for the street name. Write a function called *SetInfo* to allow for the information to be set through parameters. Write a function called *Display* that is virtual to print out the address.

- (b) (10 pts) Now write a class to represent an apartment address. Your class should be called *Apartment* and should publicly inherit the *Address* class. Write a function called *SetInfo* to allow for the information to be set through parameters. Write a function called *Display* that is virtual to print out the apartment address.

- (c) (8 pts) Assume the following functions have been added to your classes. A non-virtual function called *Print* has been added to your *Address* class to print out the address. A non-virtual function called *Print* has been added to your *Apartment* class to print out the address. Given the following declarations and initializations:

```
Address A1;
A1.SetInfo(1111, "Wood Rd.");

Apartment A2;
A2.SetInfo(3333, "Date Ave.", 202);

Address *locations[2];
locations[0] = &A1;
locations[1] = &A2;

Address A3;
A3 = A2;
```

Write the corresponding letter of the function that is called for each code segment.

- | | | | |
|-------|--------------------------|---|----------------------|
| ___ 1 | A1.Display(); | A | address::Display() |
| ___ 2 | A2.Display(); | B | apartment::Display() |
| ___ 3 | locations[0]->Display(); | C | address::Print() |
| ___ 4 | locations[1]->Display(); | D | apartment::Print() |
| ___ 5 | A3.Display(); | | |
| ___ 6 | A2.Print(); | | |
| ___ 7 | locations[0]->Print(); | | |
| ___ 8 | locations[1]->Print(); | | |