CS 7B - Fall 2018 - Final Exam

Write responses to following on separate paper. Computers may be used during the second half.

1. Explain what the following program does:

```
#include <iostream>
using namespace std;
int main() {
   int num;
   float sum=0, term=1;
   cout<<"Enter the number:";
   cin>>num;
   for(int n=num; n > 0; n /= 10 ) {
      int digit = n % 10;
      sum += digit * digit * digit;
   }
   if( sum == num )
      cout << "This is an Angstrom number" << endl;
   else
      cout << "This is not an Angstrom number" << endl;
}
```

2. This problem has three parts:
   (a) Create a `Time` class that stores a time as a single number of type `time_t` (this is just another name for a certain type of integer). The constructor should take one argument – the initial value to set the internally stored time to. This argument should have a default value of time(0) (the time function is defined in the C++ Standard Library header ⟨ctime⟩. Also create a `getter` function for the time stored in `Time` objects, and a `setter` function to allow changing the time the `Time` object stores later on.
   (b) Rewrite the constructor to use the member initializer syntax instead of an assignment statement.
   (c) Write a driver with `main()` to test your class, producing user dialog on the console like this:

```
tNow = 1544552083
The time is now 17
Enter a new value for time: 18
The new time is 18
```

3. Debug the following program, whose intent is clear enough. Include the entire fully functional program as your answer.

```
class member {
    int memberNum = 25;
    float memberPay;
public
    void Input(cin >> memberNum >> memberPay);
    void Output;
}
int main() {
    Member mem;
    cin >> mem.memberNum >> mem.memberPay;
    cout << mem.memberNum << '\t' << mem.memberPay;
}

void Output::Member() {
    cout << mem.memberNum << '\t' << mem.memberPay;
}
```

4. Fill in the blanks in each of the following statements:
   (a) A base class's________members can be accessed only in the base-class definition or in derived-class definitions.
   (b) A base class's ________ members are accessible within that base class and anywhere that the program has a handle to an object of that base class or to an object of one of its derived classes.
   (c) A base class's protected access members have a level of protection between those of public and ________access.

5. Define an `Array` class that expands the functionality of C++ arrays. Show how you would split your definition into an `Array.h` file and an `Array.cpp` file.

   The class should contain one data member that is a pointer to a dynamically allocated array of integers, and an integer that stores the current `size` of the array. The `size` integer should be `const` – it should not be changeable after the class constructor is called.

   The class should have 3 constructors: one that takes just a number of elements to allocate (initializing them all to 0), another that takes a number of elements and an array of initial elements, and a copy constructor that allocates a new array of the same size as the `Array` that is being copied, and then copies the elements one by one.

   Define 4 member functions: `getLength()` to return the number of elements in the `Array`; `getElement(int n)` to take an integer `n` and return a modifiable reference to the `(n+1)`st element of the array; another `getElement()` function that is declared `const` and returns a non-modifiable reference; and a `print()` function that takes a separator string and prints the elements one by one separated by the separator string. (For instance, `myArray.print("\n")` should print the array elements with newlines between each pair.)

   Also define a `destructor` to deallocate the memory that was allocated to the internal array when the `Array` object is destroyed.

   You do not need to define a main function that actually uses this class, but it will presumably be useful for testing purposes to try creating a few `Array` objects and manipulating them.

   Note: Problems 6,7,8 and 9 on the next page go together.

6. Write a function `createFile(ofstream& ofs, vector<string>);`to store text files and call it twice in a `main()` function to create two text files `file1.txt` and `file2.txt`. Store the following content in these two text files:

File 1:
```
The
is
the
```

File 2:
```
cat
in
hat.
```

Starter code:

```
void createFile(ofstream& ofs, vector<string> vs) {
  for (string s : vs) \\code here;
  ofs.close();
}
```

Hints: Use `ofs << s.c_str()` where `ofs` is an output file stream and `s` is a string from the string library that needs to be converted to a c-string to be inserted into the output file stream.

7. Create another function, `mergeFiles(ifstream& ifs1, ifstream& ifs2, ofstream& ofs)`, to write to a third text file `file3.txt`, which will then contain the text of `file1.txt` merged with that of `file2.txt` in the following way:

```
The cat is in the hat.
```

To do this, create 3 separate objects–2 in input mode and one in output mode.
You will also have to use conditional statements to check for the ends of files.

8. Now create `addToFile(ofstream& ofs)` to allow the the user to append some text to the end of `file3.txt` existing text file. Get user input one line at a time and query the user after each entry as to whether or not they want to append more text. The console might look like this:

```
Enter a string to add to the file: But that’s a dog’s hat!
Do you want to add some more text? Y/N: y
Enter a string to add to the file: Which can only spell trouble...
Do you want to add some more text? Y/N: y
Enter a string to add to the file: with a capital ’T’.
Do you want to add some more text? Y/N: n
```

After which `file3.txt` would contain

```
The cat is in the hat. But that’s a dog’s hat! Which can only spell trouble...with a capital
’T’.
```

9. Finally, count and print:

   (a) The number of occurrences of a user-specified letter in your file `FILE3.TXT`.

   ```
   Enter a string to add to the file: But that’s a dog’s hat!
   Do you want to add some more text? Y/N: You can’t get in that!
   Enter a string to add to the file: Do you want to add some more text? Y/N: n
   Enter a character to count in file3.txt: t
   There is/are 11 instance(s) of the character t in file3.txt.
   ```

   (b) The number of occurrences a user-specified string in your file `file3.txt` (You can use the `strcmp()` function of the `string` library here, or compare using string objects, with which you can use the `==` operator).