1) Consider the following code:

```
#include <iostream>
#include <string>
using namespace std;
void func1(int a) {
        a = a*a;
        cout << a << endl;
}
void func2(int &a) {
        a = a+a;
        cout << a << endl;
}
void func3(int *a) {
        a++;
        cout << *a << endl;
}
int main() {
        int a = 10;
        func1(a);
        cout << a << endl;
        func2(a);
        cout << a << endl;
        func3(&a);
        cout << a << endl;
}
```

a) (10pts) What would be the output of the program
b) (10pts) Explain the significance of the parameter passing for func1, func2, and func3.
c) (10pts) If we replace the `int a = 10;` with `int *b = new int;` how would we have to modify the rest of main so that the output was the same as part a
d) (5pts) Fix func3 so that it adjusts the value of a
e) (5pts) How would you pass a pointer as a parameter so that you can adjust the address.

2) You are developing an inventory system to track the number and price of items in a small store. This system will use a vector based on a dynamic array for storage. You may add additional data members and methods as you need, but justify each data member and provide code for each additional methods.

a) (5pts) Declare a `struct item` to represent each item. It should contain at least: an item name, the number of items, the price of each item.
b) (10pts) Declare a `class inventory` that keeps all data members private and all methods public. It should use a dynamic array of `item` for storage (always sized exactly to the number of items). It should at least have methods: constructor, destructor, addItem, removeItem, findTotalInventoryCost, sortItems, printInventory.
c) (15pts) Write the code for the `addItem` method. Item names are unique, so adding the same item should realistically adjust the number and price.
d) (10pts) Write the code for the `removeItem` method.
e) (10pts) Write the code for the `printInventory` method so that it prints each item (all fields), the total price of each item, and the total price of all of the inventory.
~~f) (10pts) Write the code for the `sortItems` method using a **selection sort** to sort the items by name.~~

3) Suppose you had the following function:
```
int func(int a, int *b, int &c) {
     cout << a << ", " << *b << ", " << c <<  endl;
     a++; b++; c++;
     cout << a << ", " << *b << ", " << c <<  endl; // (c) stop
}
```
   a) (10pts) Explain the significance of the parameter passing for each parameter in the function
   b) (10pts) What would be the output if your main was, explain:
```
int main() {
     int x = 10;
     func(x, &x, x);
     cout << x << endl;
}
```
   c) (10pts) Draw a diagram of the stack memory at the time the program is about to execute the second 'cout' in func. Indicate the locations of the local variables and the basic stack frames.
   d) (10pts) Explain how templates work in C++ and comment on their advantages and disadvantages.

4) As part of a Banking system, you are asked to create a Transaction Logging system to keep track of every successful transaction for individual customers. Each transaction has: transaction id (int), date (Date), time (Time), type (withdraw/deposit/transfer), source account number (int), destination account number (int), amount (double). For a withdrawal only the source account is given, for a deposit only the destination account is given, and for a transfer both are given. Assume Date and Time are available data types that offer a full range of operations (printing, comparing, assigning etc.)
   a) (5pts) Declare a `struct transaction` to represent each transaction. It should contain at least `transaction id, date, time, type, source account number, destination account number, amount`.
   b) (10pts) Declare a `class TLog` that keeps all data members private. It should use a dynamic array of `transaction` for storage (add any other members you need). It should at least have the following public methods: `constructor, destructor, withdraw, deposit, transfer, findTransactionsByDateRange, printTransactions`. Make sure you specify the parameters and return values of each method.
   c) (5pts) Write the code for your constructor.
   d) (15pts) Since `withdraw`, `deposit`, and `transfer` all add a transaction to the array, write a common private method to add a transaction of any type to the array. The transaction id starts at 1 and is automatically increased for every transaction added.
   e) (15pts) Write the code for the `findTransactionsByDateRange` method. Since date and time are always moving forward, the internal array is always sorted by date and time. This should return an array of `transaction` and its size.
   f) (10pts) FDIC regulations requires Banks to keep logs for 5 years. Explain how you would implement a `purge` method that runs at the end of every year and removes any transaction older than 5 years. Try to reuse the existing methods.