Midterm 1

All code should be written in C++. Unless otherwise specified, you may (and I generally will):

- assume that the user of any code you write will be cooperative with the input they supply;
- omit `std::` and the return value of `main()`;
- assume that all necessary libraries have been `#included`;
- omit `main()` entirely for problems that ask ONLY for function definitions;
- not concern yourself with having optimal solutions (within reason);
- not worry yourself about prompt messages for user input (I sometimes give descriptive prompts to clarify problems);
- not recopy code I have provided, or which you have written in other parts of problems.

Partial credit *will* be given, so do your best if you encounter a difficult question. PLEASE BOX YOUR ANSWER if it is not otherwise clear!

1. Consider the following code.

```
void fn(int arr[], int &x, int pos) {
   arr[pos] = x; // **Line 1
   ++x;         // **Line 2
   pos = -1;
   cout << "1) " << arr[0] << "." << x << "." << pos << endl;
}

int main() {
   int x[] = {13, 14, 15}, y = 78, z = 2;
   fn(x, y, z);
   cout << "2) " << x[0] << "." << x[1] << "." << x[2] << endl;
   cout << "3) " << y << "." << z << endl;
}
```

   a. What would this display when run?

   b. Suppose that `**Line 1` and `**Line 2` were both removed. You then consider two possibilities to replace those lines:

   `arr[pos] = x++;`

   OR

   `arr[pos] = ++x;`

   One of these changes the output from part a. Which one changes it, and how does it change?

2. Write a program which asks the user to enter single words repeatedly; you may assume that whatever the user enters does not contain spaces. The program should keep asking the user until they enter a word which ends with a period. The program should write the words the user enters into a file named `sentence.txt`.

3. Consider the following code.

```
int main() {
    double abc[] = {1.23, 4, 5};
    double *p1, *p2 = new double;
    p1 = abc;
    *p1 = 77;
    p2 = p1 + 2;
    *p2 += 8;
    p1 = p1 + 1;
    *p1 = *p2;
    cout << abc << endl;
    cout << abc[0] << "|" << abc[1] << "|" << abc[2] << "|" << p2 << "|" << &(abc[1]);
}
```

a. Suppose that the first print displays `0x1236`. What would (probably) print out from the second line? (Hint: `doubles` are typically twice as big as `ints`.)

b. Explain why, even if there were additional lines of code after the last `cout`, there would definitely be a memory leak in this code.

4. *For this problem, assume that there is a function called* `rando()` *which returns a random computer generated integer. You don't have to write this function – just call it!*

   a. I am selling user codes for access to an online platform. Each access code is a sequence of 5 ints.

   Write code that asks the user to enter an integer N. The program should then create a variable `my_codes`, which stores N access codes. Each integer in each of these N access codes should be initialized randomly, by calling `rando()`.

   Your code should work so that, for example, if I accessed `my_codes[7][2]`, it would give the 3rd number in the 8th access code.

   b. Now, write code which releases all the dynamically-allocated memory created in part a.

5. a. Write a function called *insert_front()*. This function should take an array of strings named `arr`, a string named `ins`, and an integer named `j` as arguments. This function should not return anything, but the portion of `arr` from index `0` to index `j` should be moved **one to the right**, and `ins` should be inserted at index `0`.

   For *example*, if

   ```
   string x[] = {"A", "B", "C", "x", "yy", "zzz"};
   ```

   then after the call `insert_front(x, "Hello", 2)`, the contents of `x` should be

   ```
   {"Hello", "A", "B", "C", "yy", "zzz"};
   ```

   You may assume that `j + 1` is less than the length of the array; and you do not need to worry about what happens to the entry originally in position `j + 1`.

   (Hint: go backwards!)

   b. I am arranging a family picture. There are 6 people in my family. I line them up as they arrive, with each successive person added to the right end of the row. Of course, some of my family members are difficult, and they just INSIST on being added to the left instead.

   I write some code to arrange my family in the picture:

   ```
   int main() {
       string picture[6];
       ...
   ```

   Write code which asks the user to input 6 names, and asks each one whether they want to be placed on the left. If so, you should place them at the beginning of `picture`; if not, you should place them directly after the rightmost name currently stored in `picture`.

   For full credit, you may only use ONE loop in `main()`, and you must call the function from part a!

   EXAMPLE: if Alice comes first; then Bob comes; then Carol comes AND she asks to be on the left then `picture` will be in the state

   ```
   {"Carol", "Alice", "Bob", "", "", ""}
   ```

   Continuing: if David comes next; then Evan comes AND he asks to be on the left; and finally Frank comes, then the final state would be

   ```
   {"Evan", "Carol", "Alice", "Bob", "David", "Frank"}
   ```

6. Write the definition of a **recursive** function `max_prod(int arr[], int i, int j, int num)` which finds the maximum product of `num` entries from `arr` (between indices `i` and `j`, inclusive). For example, if

   ```
   int x[] = {4, 11, 3, 10, 5, 2, 1};
   ```

   then `max_prod(x, 0, 6, 3)` would be 550, since the maximum product of three elements from the list is 11 times 10 times 5.

   Hints: the maximum product either includes the first element (and some elements from the rest), or it includes only elements from the rest.

   Also, there are two base cases you might want to think about: where `num` is `0`, and where `num` is larger than the length of the range in the array that you want to look through.

7. I have a file named `shopping.txt`, which contains several sentences, which looks like:

```
I buy 5 computers.
I buy 13 books about programming.
I buy 25 used keyboards.
```

The file could have several lines, but each sentence will be on a line by itself, and start with "I buy" followed by a number, and an item.

Write a program which opens this file, and prints out the total number of items I buy. (For example, with the file shown, the number printed would be 43, since $5 + 13 + 25 = 43$.) I suggest using stringstreams.

For full credit, your program should work for any file named `shopping.txt`, which contains some number of sentences which begin with "I buy" followed by an int.