5.

On my computer, I've got two grade files. One contains the grades from the first mid-term (`mt1.txt`), and the other contains the grades from the second mid-term (`mt2.txt`).

Each file has exactly the same format: a lot of lines each containing the name of the student (they are all quite famous, so they all have just one single short name) followed by four integer grades. Plus, the lines are all in alphabetical order.

Here is a very small example of the two files (the real ones are quite long):

<u>mt1.txt</u>

```
    Bambi 25 13 30 0
    Cher 11 21 17 1
    Frank 31 25 26 1
    Madonna 3 7 0 1
    Zelda 19 28 25 1
```

<u>mt2.txt</u>

```
    Bambi 31 27 17 0
    Cher 26 19 21 1
    Frank 33 20 24 1
    Madonna 0 5 0 0
    Zelda 33 29 33 1
```

As you can see, the two files have grades for the same students in the same order.

A. Write a program that would read two files just like these (but longer) and combine them into a single file containing one line for each student, in the same order, with that one line now containing all eight of the student's grades. For the example above, the new file would be:

<u>allmt.txt</u>

```
    Bambi 25 13 30 0 31 27 17 0
    Cher 11 21 17 1 26 19 21 1
    Frank 31 25 26 1 33 20 24 1
    Madonna 3 7 0 1 0 5 0 0
    Zelda 19 28 25 1 33 29 33 1
```

Write very clearly and leave yourself a lot of space, because for part B, you will be modifying this program.

Try to make it a whole program, with the correct C++ #includes, not library.h. Although this is not a major requirement.

Most of the points are for part A.

B. Sometimes students are frightened by the first mid-term and drop the class. This means that the second file might have some people missing from it. The files could be something like this:

<u>mt1.txt</u>
```
    Bambi 25 13 30 0
    Cher 11 21 17 1
    Frank 31 25 26 1
    Madonna 3 7 0 1
    Zelda 19 28 25 1
```
<u>mt2.txt</u>
```
    Bambi 31 27 17 0
    Frank 33 20 24 1
    Zelda 33 29 33 1
```

But at least the two files will still be in alphabetical order.

Show the modifications to your program that would be required to handle dropped students properly. When a student does not appear in the second file, the new file should still show their grades from the first mid-term. So for the new example, the output would be:

<u>allmt.txt</u>
```
    Bambi 25 13 30 0 31 27 17 0
    Cher 11 21 17 1
    Frank 31 25 26 1 33 20 24 1
    Madonna 3 7 0 1
    Zelda 19 28 25 1 33 29 33 1
```

6.

An automatic animal monitoring device sits in the african savannah monitoring all the animals that pass within range. Throughout the day it creates a large text file that records all of its observations. At the end of the day that file is uploaded to a research base and analysed.

The file has one line for each animal observed. Each line has the same format: first the time (in 24 hour form), then the species of the animal (a single word, using only lower case letters), then the direction it was heading (N, S, E, or W), then an estimate of its weight (in pounds), and finally an estimate of its speed (in miles per hour). Here is a sample from a typical file...

```
0530  gorilla  E  441.3  2.9
0531  gorilla  E  338.0  2.8
0531  gorilla  E  127.5  2.8
0540  zebra  W  1107.0  11.0
0542  lion  W  510.3  15.0
0551  penguin  S  5.1  45.0
0551  gorilla  W  338.0  2.1
```

It seems to show a family of gorillas strolling towards the rising sun, then shortly later a zebra chased by a lion going in the opposite direction, and then a small penguin flying south and one of the gorillas coming back. The name of the file is `animals.txt`.

Write a function that processes this file and does the following things:

+ It must create a new file called `gorillas.txt` containing all the records of gorilla observations.
+ It must calculate the average weight of all the lions observed.
+ It must note the speed of the slowest gorilla seen heading east in the morning (time between 0000 and 1200).

So if the sample data given above were the whole file, your function would create a `gorillas.txt` file containing

```
0530  gorilla  E  441.3  2.9
0531  gorilla  E  338.0  2.8
0531  gorilla  E  127.5  2.8
0551  gorilla  W  338.0  2.1
```

And it would print for the user to see:

```
Average lion weight 510.3 pounds
Slowest east-bound morning gorilla 2.8 mph
```

7.

A. Write a program to print a multiplication table, using **for** loops.

Accept the size of the multiplication table from the user. Include relevant validations, to discard inputs that are irrelevant (less than 1 and greater than 20)

If the input is 5, the output should print the following

```
  1  2  3  4  5
  -------------
1 1  2  3  4  5
2 2  4  6  8 10
3 3  6  9 12 15
4 4  8 12 16 20
5 5 10 15 20 25
```

B. Write a program using **while** loops, to print the sum of all digits of a positive integer.

Input : 2345
Output: 14

Input: 5930
Output: 17

C. Write a program using while loops, compare whether two strings are same.

Hint: You can use length() to get the size of string.
For example, `string x = "hello";  x.length()` will return 5.

8.

Write three versions of program to convert a decimal number to a binary number. The program should accept the input decimal number from the user.

Example Input: 33 Output: 100001

a. Version 1: Binary converter using recursion

b. Version 2: Binary converter using while loops

c. Version 3: Binary converter using for loop.