EEN218
A Grand Old Test
5th March 2013

Who are you?

What is your student number?

Did you cheat on this test?

Sign that statement.

Don't make any marks in my boxes.
These are my boxes.

Question	1	2	3	4
Out of	33	33	33	1
Grade				

1

This is a function that multiplies together two square matrices, it requires that each matrix has the same number (N) of rows and columns.

```
matrix * multiply(matrix * A, matrix * B, int N)
{ matrix * C = new matrix(N, N);
  for (int row = 0; row < N; row += 1)
  { for (int col = 0; col < N; col += 1)
    { double sum = 0.0;
      for (int i = 0; i < N; i += 1)
        sum += A->get(row, i) * B->get(i, col);
      C->set(row, col, sum); } }
  return C; }
```

The matrix methods used should be obvious:

`new matrix(A, B)` creates a new A×B matrix (A rows, B columns)
`get(r, c)` provides the value stored at row r, column c
`set(r, c, new_value)` changes the value stored at row r, column c

a. In terms of linear, quadratic, logarithmic, and so on, what is the speed of this function. Consider N to be the data size.

b. Write that using the Big-O notation.

c. If it takes 0.5 seconds for that function to multiply together two 100×100 matrices (i.e. when N=100), how long would you expect it to take to...
   i. multiply together two matrices when N=200?
   ii. multiply together two matrices when N=1000?
   iii. multiply together two matrices when N=10?

d. (corrected)
N isn't really a good measure of the data size for a matrix, when N=20, there are 400 numbers. That would be a much better measure. Answer part ~~ii~~ b again, this time using this more accurate notion of data size.

This is the function for adding together two square matrices.

```
matrix * add(matrix * A, matrix * B, int N)
{ matrix * C = new matrix(N, N);
  for (int row = 0; row < N; row += 1)
  { for (int col = 0; col < N; col += 1)
    { double val = A->get(row, col) + B->get(row, col);
      C->set(row, col, val); } }
  return C; }
```

e. If on another computer it takes 1 milli-second for that function to add together two 100×100 matrices (i.e. when N=100), approximately how long would you expect it to take to...

  i. add together two matrices when N=200?
  ii. add together two matrices when N=1000?
  iii. <u>multiply</u> together two matrices when N=100?

2

The Farm Animals' Union wants a database for its membership lists. A basic struct to represent a single farm animal has already been defined, and includes the following public members

`string name;`	the animal's name
`string farm;`	the name of the farm that it lives on
`int mem_num;`	the animal's union membership number
`int paid;`	the date the animal last paid its membership dues.

a.

Implement the database in the form of a vector, but don't #include `<vector>`. The database should have:

- a constructor,
- a destructor,
- an add method for adding a new animal,
- a remove method to be used when a member is sold or eaten,
- any other methods that make your task easier.

How you implement remove is for you to decide, but however you do it, it must not leave a gap in the database.

For all parts of the design, use best programming practice, not laziness. That means, among other things, that you should use public and protected appropriately.

b.

When comparing two farm animals in order to put them in an alphabetical list, first their farms are checked. If two animals live on the same farm, then the comparison is based on their own names. If they live on different farms then the comparison is based on their farm's names. To illustrate, these animals are all in the correct order

Colin the Cow, of Lazy-E ranch
Zelda the Cow, of Lazy-E ranch
Bossy the Cow, of McDonald's farm
Mooey the Cow, of McDonald's farm
Smellie the Cow, of McDonald's farm
Abigail the Pig, of Zeke's farm
Oinky the Pig, of Zeke's farm

Write a method that will cause the entire database to be sorted in this way. You can use any sorting algorithm you want: no need for anything very fast.

3

Write a function that would partition an array of strings, so that it could be used as part of a quicksort implementation.

This should be the function's prototype:

```
     int partition(string A[], int begin, int end);
```

it must choose a pivot, partition the portion of the array between `A[begin]` and `A[end]` inclusive, and return as its result the position of the end of the left partition.

That is, if `partition`(A, begin, end) returns X when it is finished, then everything from A[begin] to A[X] should be less than or equal to the pivot, and everything from A[X+1] to A[end] should be greater than the pivot.