EEN118
22nd November 2011

No Electronic Devices
No Books or Notes
No Panicking

Name:

Student number:

Sign here if you did not give or receive aid in any form during this test, and did not consult any written or printed material apart from this test:

Don’t write in these boxes, they are mine.

Question	Out of	Grade
5	33	
6	33	
7	33	
8	1	

5.

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
```

6.

A struct (or object) to represent dates is to be implemented. It must contain a string for the day of the week, and three numbers for the day of the month, the month, and the year.

a.

Give the struct definition, along with a suitable `set` function for initialising a date, and a `print` function that prints one nicely. Using your definitions, I should be able to write this:

```
date today, st_swithins_day;
set(today, "Tuesday", 22, 11, 2011);
set(st_swithins_day, "Friday", 15, 7, 2011);
print(today);
```

and when run, that code should print something very close to this

```
Tuesday 22nd November 2011
```

or this

```
Tuesday November 22nd 2011
```

b.

Define a function called `latest`, which takes two `date`s as parameters, and returns as its result the one that is latest (i.e. comes second). So for example, this

```
date x = latest(today, st_swithins_day);
print(x)
```

should also print today's date.

c.

Define a function called `next`, which takes one `date` as its parameter, and modifies that date by moving it on to the next day, so this

```
next(today);
print(today);
next(today);
print(today);
```

would print

```
Wednesday November 23rd 2011
Thursday November 24th 2011
```

7.

Here is an attempt at sorting an array of ints so that they will appear in ascending order. It consists of a helper function that finds the largest int in an array, and a sorting function that repeatedly finds the largest int and moves it to the end of the array. Finally, there is a main() that tests it.

```
int find_biggest(int A[1000])        Don't say the 1000      add , int N
{ int biggest_so_far = 0;
  int pos = 0;
  while (pos<=1000)                  < not <=        N instead of 1000
  { if (A[pos]>biggest_so_far)       A[pos] > A[bigg.s.f]
      biggest_so_far = A[pos];       not A[pos], just pos
    pos = pos + 1; } }                        return something   biggsest.s.f

int swap(int x, y)             int y not just y          also must be & x and & y
{ int t = x;
  y = t;                 t = x; x = y; y = t;
  x = y; }

void sort(int A[1000])         son't say 1000      add, int N
{ int end_pos = 1000;        start at 999      no, N - 1 instead
  while (end_pos>0)          must be >=
  { int big = find_biggest(A);       change to A, end_pos
    swap(A[big], A[end_pos]); } }      also end_pos = end_pos - 1

void main()
{ int data[1000];
  int number;
  cout << “How many numbers are you going to type? ”;
  cin >> number;
  cout << “OK, now type them all\n”;
  for (int i = 0; i<number; i += 1)
    cin >> data[i];
  sort(data[1000]);            sort(data, number);
  cout << “Here they are sorted\n”;
  for (int i = 0; i<number; i += 1)
    cout << data[i] << “\n”; }
```

There is a lot wrong with this program.

Tell me all the mistakes, and what must be done to fix them and make it work.

To make it easier on you, I’ll tell you that there is nothing wrong with the input and output part, so don’t waste time looking for mistakes in the parts with grey backgrounds.

8.

Draw a picture of a cat with a hat, or a hat with a cat, whichever you prefer. The cat's name is Dennis.