1.

a. A mysterious data processing program is being analysed, and one of the tests being performed is to give it various amounts (N) of random input data, and measuring how long it takes (T) to complete the processing. These are the results:

input size, N	processing time, T
400	0.289 S
500	0.540 S
700	1.240 S
1000	4.365 S
1300	9.233 S
1700	20.944 S

- Using the accompanying log-log graph paper, determine what the speed of this algorithm is (in terms of linear, logarithmic, cubic, quadratic, etc).

- How long would you expect it to take to process 2,000 data items?
- How long would you expect it to take to process 20,000 data items?
  Use units of time that make the answer immediately comprehensible to an ordinary person.

- Write a formula for calculating the approximate value of T from a given value of N.
  It should be of this form:

$$T \approx \textit{some-formula-involving-}N \text{ Seconds}$$

b. Explain exactly why selection sort is Quadratic in time.
This should not be a long answer, but should be complete. You should not assume that the reader knows the selection sort algorithm, but he or she does know the basics of programming, and does know what quadratic means.

c. If a particular program is known to be Quadratic, and it takes 10 seconds to process 10,000 data items, how long would you expect it to take to process:
- 20,000 data items?
- 1,000,000 data items?
- 1,000 data items?

try to use sensible (for ordinary humans) units of time.

d. If a particular program is known to be Cubic, and it takes 10 seconds to process 10,000 data items, how long would you expect it to take to process:
- 20,000 data items?
- 1,000,000 data items?
- 1,000 data items?

try to use sensible (for ordinary humans) units of time.

2.

A struct for representing a company's employees has already been defined thus:

```
struct person
{ string name;
  string department;
  double salary;

  person(string n, string d, double s)
  { name=n; department=d; salary=s; }

  void print()
  { cout << name << ", " << department << ", $" << salary << "\n"; } };
```

we want to create a linked list for storing a whole lot of them.

a. Define the necessary struct(s) or class(es),
Write a function suitable for adding a new person the front of the list, and
Write a function that is given a person's name, and searches the list for that person, printing their information if found.

Using your code, this piece of program should work:

```
personlist P;
add(P, new person("Marmaduke Sneem", "accounts", 43500.00);
add(P, new person("Bumbellina Hairsmith", "hexagons", 64910.00);
add(P, new person("Oswaldo McDuck", "accounts", 40250.00);
add(P, new person("Tyrannosaurus Jones", "advertising", 54710.00);
find_and_print(P, "Oswaldo McDuck");
```

If you prefer, `add` and `find_and_print` may be written as methods instead of regular functions.

b. Write another function (or a method, your choice) that finds the employee with the highest salary in a linked list, and returns their information as its result.

c. Design and write another function (or a method) that finds the employee with a given name (like in part a), and removes him or her from the list.

Think carefully about your plan before you start coding. Work out what you have to do in the normal case (when the list contains a moderate number of entries and the employee to be removed is somewhere in the middle). Then think about the possible special cases: will your method still work when it has to remove the last employee in the list? will it work for the first employee in the list?

Write the plan for your function in plain English (not too wordy, just say how it will work) as part of your answer.