Sample Question 1.

A. Write a function that takes at least two parameters: a string S, and an array of strings A. It should search through A to see if S appears anywhere in it. If S is found, the function should return its position (remember that the first position in an array is numbered 0). If S is not found, the function should return -1.

so in this context

```
void main()
{ string names[] = { "Jilly", "Joe", "Jim", "Jenny", "Jemimah" };
  int p = find("Jenny", names,...);
```

p should be set to 3.

B. Write another function that takes at least two parameters: an integer position P, and an array of strings A. It should change the array A by "removing" the string which is at position P.

Of course, you can't just leave a hole in an array. Removing a string is achieved by moving all following strings up by one position so that the unwanted one is overwritten, and noting that the array is now a little shorter than it used to be.

so after doing this

```
void main()
{ string names[] = { "Jilly", "Joe", "Jim", "Jenny", "Jemimah" };
  remove(2, names,...);
```

names should contain "Jilly", "Joe", "Jenny", and "Jemimah".

C. Making use of your answers to parts A and B, write another function that takes two arrays of strings. It should modify the first array by removing from it any string that appears anywhere in the second array.

Think before writing. There is a very easy way to do this.

Sample Question 2.

A. A particular club whose members are interested in cats wants to computerise all their records. For each cat they record its name and colour (both strings), and its number of legs, weight, and value (all ints).

+ Design a struct or object suitable for representing cats.
+ Provide a set function for initialising a cat object with those five pieces of information, and
+ Provide a suitable print function for displaying the information for a cat nicely.

B. It is in fact a club only for very special people who own exactly three cats, no more, no less. Every member has a name, and address, and three cats.

+ Design a struct or object suitable for representing members.
+ Provide a set function for initialising a member object with those five pieces of information, and
+ Provide a suitable print function for displaying the information for a club member nicely.

C. For entertainment, the club members sometimes set up fights between their cats (I said they were *special* people, not *nice* people). They have discovered a few things:

+ A cat with 4 legs always defeats a cat with some other number of legs.
+ If that does not settle the matter, heavier cats always beat lighter cats.
+ If that leaves the result unresolved, cheaper cats always beat more expensive cats.
+ After that, the winner is just random.

Write a function that takes two cat parameters and determines the outcome of a fight between them. It should return 1 if the first cat parameter wins, and 2 if the second wins.

D. The club members also have contests against each other. When member A battles member B, this is how it works. They make A's first cat fight B's first cat, then they make A's second cat fight B's second cat, then they make A's third cat fight B's third cat.

Whoever's cats win most times (it can't be a tie because they all have three cats) is the winner. The defeated member has his name officially changed to "Mr. Loser".

Write a function that carries out the battle between two club members.

Sample Question 3.

What is the output of following program ?

```
#include <library.h>

void main()
{
     int sum = 40, prod = 1;
     int a[2][3][4] = {{
                         {8, 3, 6, 4},
                         {9, 9, 4, 8},
                         {5, 5, 7, 8}
                         },
                         {
                         {7, 8, 4, 5},
                         {1, 3, 8, 2},
                         {8, 8, 8, 9}
                         }};

     int b[3][4] = {0};

     cout<<"1a: "<<a[1][2][3]<<endl;

     cout<<"1b: "<< b[2][3]<<endl;

     for(int i = 0; i<2; i++)
          for(int j = 0; j < 3; j++)
               for(int k = 0;k < 4; k++)
                    b[j][k] += a[i][j][k];


     cout<<"2: "<< b[2][1]<<endl;


          for(int x = 0; x < 3; x++)
               for(int y = 0;y < 4; y++)
                    if( (x+y) % 2 == 0)
                         sum += b[x][y];

     cout<<"3: "<< sum<<endl;

     for(int p = 0; p < 3; p++)
               prod *= b[p][p];

     cout<<"4: "<< prod<<endl;
}
```

Sample Question 4.

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, "Friday", 20, 4, 2018);
set(st_swithins_day, "Friday", 15, 7, 2018);
print(today);
```

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

```
Friday April 20th 2018
```

or this

```
Friday 20th April 2018
```

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);
```

would print

```
Saturday April 21st 2018
Friday April 20th 2018
```

Sample Question 5.

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])
{ int biggest_so_far = 0;
  int pos = 0;
  while (pos<=1000)
  { if (A[pos]>biggest_so_far)
      biggest_so_far = A[pos];
    pos = pos + 1; } }

int swap(int x, y)
{ int t = x;
  y = t;
  x = y; }

void sort(int A[1000])
{ int end_pos = 1000;
  while (end_pos>0)
  { int big = find_biggest(A);
    swap(A[big], A[end_pos]); } }

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]);
  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.