Problem 1 a *(10 points)* Write a complete C++ program that asks the user to enter their age and the number of pets that they have at home. A legal age must be between 1 and 100 (inclusive).

If the user enters an illegal age the program should print `I don't believe you!`

Otherwise if the number of pets is divisible by the age (without a remainder) the program should print `That is a lot of pets.`

Here is a sample to show how the program runs.

```
Enter the your age and number of pets:  20 200
That is a lot of pets.
```

Answer:

```
#include <iostream>
using namespace std;

int main() {
   int age, pets;
   cout << "Enter the your age and number of pets:  ";
   cin >> age >> pets;
   if (age < 1 || age > 100)
      cout << "I don't believe you!\n";
   else if (pets % age == 0)
      cout << "That is a lot of pets.\n";
   return 0;
}
```

```
Award partial credit for the following elements of a program:

2 point for overall structure: includes, main etc.
1 points for declaring the 2 variables.
1 point for prompt.
1 point for input of 2 variables.
2 points for the condition of the first if.
  (1 for the || and 1 for the 2 parts).
2 points for the else if and condition
1 point for correct output including newlines.

If a program follows a different (but reasonable) plan. Try to award
partial credit for meeting similar goals in the program.

If you feel that a solution is too long and messy to grade properly
(longer than about 25 lines if they were normally spaced out)
judge a rough score for partial credit by the above milestones and
then adjust the score down by the following guidelines.

Very long and messy but looks probably correct --- max allowed score is 8/10.
Very long and messy and looks partially correct --- max allowed score is 6/10.
Very long and messy and looks probably badly wrong --- max allowed score is 4/10.
```

Problem 2 a *(10 points)* Consider the following C++ program. The program makes use of a function `first3digits` that returns the number formed by the first 3 digits of its input argument as its result. So for example, `first3digits(12345678)` would be `123`.

Make sure to use your own 8-digit CUNY ID number as the number entered as input to the program. It would be a very bad idea to give answers based on another student's ID number!

```
int main() {
   int id, n, x = 100, y = 17, z = 19;

   cout << "Enter your 8-digit CUNY id number: ";
   cin >> id;     // assume that the user types YOUR OWN CUNY ID number

   cout << id << endl;                                          // line (a)
   n = first3digits(id);

   cout << n % x << endl;                                       // line (b)
   cout << n / x << endl;                                       // line (c)

   if ( (y < z)  && ((x % y) < y) ) cout << "Yes\n";            // line (d)
   else cout << "No\n";
   y += 3; z /= 3;
   cout << y << z << y << endl;                                 // line (e)

   return 0;
}
```

(a) What is the output from the instruction beginning on line (a)?

Answer:

```
12345678

This answer is based on the ID number 12345678.  Actual answers will be
different.
```

(b) What is the output from the instruction beginning on line (b)?

Answer:

```
23

The answer will be made from the 2nd and 3rd digits of the answer to (a).
```

(c) What is the output from the instruction beginning on line (c)?

Answer:

```
1

The answer will be the 1st digit of the answer to (a).
```

(d) What is the output from the instruction beginning on line (d)?

Answer:

```
Yes
```

(e) What is the output from the instruction beginning on line (e)?

Answer:

```
20620
```

```
2 points per part.  One point partial credit for answers that are almost
correct except for a tiny error.  (A tiny error might be extra or missing
new lines or spaces.)

Check carefully that the 8 digit CUNY ID number is correct.  If not
email me what would be correct and what was used.

Note that the answers to parts b and c depend on the ID number.
```

Problem 3 a *(10 points)* Write a complete C++ program that repeatedly asks the user to enter a number of `rows`. If `rows` is greater than or equal to 0, the program prints a triangular pattern of `*s` with that number of rows. When a user enters a negative number of rows, the program tells the user the total number of `*s` that have been printed and terminates.

Here is a sample to show how the program runs.

```
Enter the number of rows or a negative number to stop: 4
*
**
***
****
Enter the number of rows or a negative number to stop: 2
*
**
Enter the number of rows or a negative number to stop: 3
*
**
***
Enter the number of rows or a negative number to stop: -1
A total of 19 *s  were printed.
```

Answer:

```
#include <iostream>
using namespace std;

int main() {

   int rows = 0, total = 0;
   while (rows >= 0) {
      for (int r = 1; r <= rows; r++) {
         for (int c = 1; c <= r; c++) {
             cout << "*";
             total++;
         }
         cout << endl;
      }
      cout << "Enter the number of rows or a negative number to stop: ";
      cin >> rows;
   }
   cout << "A total of " << total << " *s  were printed.\n";
   return 0;
}
```

```
Award partial credit for the following elements of a program:

1 point for overall structure: includes, main etc.
1 point for correct declaration and initialization of two variables.
1 points for a correct outer loop (likely while)
2 points for correct nested inner loops (likely for)
1 point for correctly printing *s
1 point for correctly printing lines
2 points for correctly incrementing total (there are many ways to place this)
1 point for correct output of total after all loops.

If a program follows a different (but reasonable) plan. Try to award
partial credit for meeting similar goals in the program.

If you feel that a solution is too long and messy to grade properly
```

(longer than about 25 lines if they were normally spaced out)
judge a rough score for partial credit by the above milestones and
then adjust the score down by the following guidelines.

Very long and messy but looks probably correct --- max allowed score is 8/10.
Very long and messy and looks partially correct --- max allowed score is 6/10.
Very long and messy and looks probably badly wrong --- max allowed score is 4/10.

Problem 4 a *(10 points)* The following program asks the user to enter a number *n*. It then prints a picture showing a triangle that points to the right that has *2n - 1* rows and n columns. The odd numbered rows are made of `*`s and the even ones are made of `o`s. For example, if *n = 4* the program would print:

```
*
oo
***
oooo
***
oo
*
```

Some pieces of code have been replaced by `PART (a)`, `PART (b)`, and so on. To answer the parts of this question you should supply the `C++` code that was replaced. Each answer must fit on a single line.

```
int main() {
   int n;
   cout << "What is n? ";
   PART (a)
   for (int r = 1; PART (b); r++) {
      for (int c = 1; PART (c); c++) {
         if (PART (d)) cout << "*";
         else cout << "o";
      }
      PART (e)
   }
   for (int r = n - 1; PART (f); r--) {
      for (int c = 1; PART (g); c++) {
         if (PART (h)) cout << "*";
         else cout << "o";
      }
      PART (i)
   }

   return 0;
}
```

(a) Give a replacement for PART (a) to read the user's value of n

Answer: PART (a) is `cin >> n;`

(b) Give a replacement for PART (b) to loop over the upper rows of the picture:

Answer: PART (b) is `r <= n`

(c) Give a replacement for PART (c) to loop over columns of the row:

Answer: PART (c) is `c <= r`

(d) Give a replacement for PART (d) to test whether to print a star

Answer: PART (d) is `r % 2 == 1`

(e) Give a replacement for PART (e) to finish each row

Answer: PART (e) is `cout << endl;`

(f) Give a replacement for PART (f) to loop over the lower rows of the picture:

Answer: PART (f) is `r >= 1`

(g) Give a replacement for PART (g) to loop over columns of the row:

Answer: PART (g) is `c <= r`

(h) Give a replacement for PART (h) to test whether to print a star

Answer: PART (h) is `r % 2 == 1`

(i) Give a replacement for PART (i) to finish each row

Answer: PART (i) is `cout << endl;`

1 bonus point for anyone who does anything on the problem.

Then 1 point per part.

This makes a total of 10.

Do not penalize students who write the whole line including the required missing part.