Group Assignment: **Due in our next class session (August 12 / 13)**

Important Notes:

1. There can be **at most 3** members per group.
2. All items in Part A should be answered.
3. Choose any two (2) items in Part B to answer.
4. Per group, there are a total of 6 programs to submit. Submission should be the printout of the source code. Bring the softcopy as well. Each file should start with an introductory comment which indicates the names of the members, with the person who created /was in-charge of the item underlined (underline is obviously handwritten). Also include a short description (or use the title) of the problem in the introductory comment.

Part A.

1. Write a program that will display the given number with commas. For example: if nNum is 12345, the program should display 12,345. Note that although the given nNum is always non-negative, it is possible for nNum to be less than 4 digits or even more than 6 digits.

2. Write a program that will display a histogram of each of the digits in a given number. For example: if nNum is 726, the displayed result on the screen should be:

```
7 *******
2 **
6 ******
```

3. Write a program that will display the following pattern given n, where n represents the number of lines and the width of the pattern to be generated. Assume that n will be a positive odd integer that will at least be 3. Your program should follow the skeleton below:

```
/* This function should display the character stored in variable c,
nCount number of times */
void display (char c, int nCount)
{
    /* provide your variable declaration and code */
}

/* This function should display nCount digits starting from *pNum
(i.e., if *pNum is initially 3 and nCount is 4, the function displays
3456 and *pNum will be 7 before the end of the function. */
void displayNumbers (int *pNum, int nCount)
{
    /* provide your variable declaration and code */
}
```

```
/* This function should display a pattern based on the given n.
```

If n is 5	If n is 9	If n is 13
XXXXX 1XXX2 34X56 7XXX8 XXXXX	XXXXXXXXX 1XXXXXXX2 34XXXXX56 789XXX012 3456X7890 123XXX456 78XXXXX90 1XXXXXXX2 XXXXXXXXX	XXXXXXXXXXXXX 1XXXXXXXXXXX2 34XXXXXXXXX56 789XXXXXXX012 3456XXXXX7890 12345XXX67890 123456X789012 34567XXX89012 3456XXXXX7890 123XXXXXXX456 78XXXXXXXXX90 1XXXXXXXXXXX2 XXXXXXXXXXXXX

```
 */
void displayPattern (int n)
{   int nLine, nNum, nX, nDigits;  /* do not add other variables */

    nNum = 1;
    nX = n;
    nDigits = 0;
    for (nLine = 1; _____)
    {
         displayNumbers(_____);

         display(_____);

         displayNumbers(_____);

         printf("\n");

          /* provide other necessary code here */
    }
 }

int main()
{   int n;

    printf("Enter n: ");
    scanf("%d", &n);
    displayPattern(n);
    return 0;
}
```

4. Write a program that accepts as input an integer in the range of 0 to 255, then outputs its corresponding 8-digit binary equivalent without leading 0's. Example: if decimal number is 19, the output is 10011.

Part B. ACM Problem Set

Problem #1: Digit Prime

(Adapted from: http://acm.uva.es/p/v105/10533.html)
Filename : problem1.c

A prime number is a positive number, which is divisible by exactly two different integers. A digit prime is a prime number whose sum of digits is also prime. For example the prime number **41** is a digit prime because **4+1=5** and **5** is a prime number. Seventeen (**17**) is not a digit prime because **1+7=8**, and **8** is not a prime number. In this problem your job is to find out the number of digit primes within a certain range less than **1000000**.

Input

The user's first input is a single integer **N (0<N<=500000)** that indicates the total number of times the program will execute. Afterwards, on every prompt of the program, two integers **t1** and **t2 (0<t1<=t2<1000000)** will be entered**.**

Output

A single integer that indicates the number of digit primes between **t1** and **t2** (inclusive).

Sample Input and Output

```
3
10 20
Output: 1

10 100
Output: 10

100 10000
Output: 576
```

Problem #2: Automorphic Numbers

(Adapted from: http://acm.uva.es/p/v104/10433.html)
Filename : problem2.c

Recreational mathematics is mathematics that is fun and used as either as a diversion from serious mathematics or as a way of making serious mathematics understandable or palatable. These are the pedagogic uses of recreational mathematics. They are already present in the oldest known mathematics and continue to the present day. One of the topic of the recreational mathematics is the **Automorphic numbers** which have many use in different fields.

Automorphic numbers are those numbers whose **square ends with the number itself**:

$5^2 = 25,$
$25^2 = 625,$
$76^2 = 5776.$

The first automorphic numbers are 1, 5, 6, 25, 76, ... By definition 0 is not an Automorphic Number and for this particular problem we will not consider 1 as an Automorphic Number.

Automorphic numbers have enormous application in mathematics. In this problem you are to determine whether a given number is Automorphic or not.

Input

User enters an arbitrarily big integer. Input ends when user enters 0.

Output

If the input is an Automorphic number print "`Automorphic number of` **`n`**`-digit.`", where **n** is the number of digits of the integer. Otherwise print "`Not an Automorphic number.`"

Sample Input and Output

```
5
Automorphic number of 1-digit.

-76
Automorphic number of 2-digit.

34
Not an Automorphic number.

0
--End of program--
```

Problem #3: Fibinary Numbers

(Adapted from: http://acm.uva.es/p/v7/763.html)
Filename : problem3.c

The standard interpretation of the binary number **1010** is 8 + 2 = 10. An alternate way to view the sequence "**1010**" is to use Fibonacci numbers as bases instead of powers of two. For this problem, the terms of the Fibonacci sequence are:

1, 2, 3, 5, 8, 13, 21, …

where each term is the sum of the two preceding terms (note that there is only one 1 in the sequence as defined here). Using this scheme, the sequence "**1010**" could be interpreted as **1** x 5 + **0** x 3 + **1** x 2 + **0** x 1 = 7. This representation is called a Fibinary number.

Input

The input to the program is 5 Fibinary numbers.

Output

The outputs are the corresponding value of the Fibinary numbers in decimal representation (base 10).

Sample Input and Output

```
100100
Output: 16

10100
Output: 11

101011
Output: 21

100000
Output: 13

11100001101
Output: 297
```

Problem #4: Ugly Numbers

(Adapted from: http://acm.uva.es/p/v1/136.html)
Filename : problem4.c

Ugly numbers are numbers whose only prime factors are **2**, **3**, or **5**. The sequence below shows the first **11** ugly numbers.

1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, …

By convention, **1** is included. Your goal is to write a program that displays the ***n***th ugly number.

Input

The user enters a positive number, ***n***.

Output

The output consists of a single line displaying the ***n***th ugly number in the sequence.

Sample Input and Output

```
10
Output: 12
```