1. *Counting Same Remainder*. Given a sequence (in non-increasing or non-decreasing order) defined by two integers ***m***, ***n***, you are to write a Java program that prints the number of integers from the sequence that have the same remainder ***r*** when divided by ***d***. (You may assume that ***r*** and ***d*** are positive integers and that ***m*** and ***n*** are themselves the numbers on the sequence).

Input: the first line gives an integer ***k*** which is the number of test cases and is followed by ***k*** lines of inputs, each of which gives the ***m, n, d, r*** for a test case (as defined above).

- Example:

```
4
1  5  2  1
5  1  2  1
30  30  1  0
2  21  3  2
```

Output: ***k*** lines of output, each of which is a single integer (for each test case) representing the result (as defined above).

- Example (the output for the above inputs):

```
3
3
1
7
```

2. *Decoding Numbers*. A country adopts a flexible number system: they use English characters (upper case) for their decimal number system; every ten years they update the first "digit" (like the "0" in our decimal system). For example, in the last 10 years, they have used "A", "B", "C", …, "J" for 0, 1, 2, …, 9; this year, they are using "B", "C", "D", …, "K" for 0, 1, 2, …, 9., …, and etc. Negative number starts with "-" (like ours), for example, -ABED. You are now to write a Java program to decode their numbers and print the corresponding numbers in our decimal number system.

Input: the first line gives an integer ***k*** which is the number of test cases and is followed by ***k*** lines of inputs, each of which gives two inputs (like strings) ***numString***, **firstDigitChar (only a single character)**, separated with a single white space.

- Example:

```
3
ABCD  A
-ABCD  A
MNOPQL I
```

Output: ***k*** lines of output, each of which is a single integer (for each test case) representing the result (as defined above, ***no*** leading **0**'s should be outputted).

- Example (the output for the above inputs):

```
123
-123
456783
```

3. *Encoding*. The CS department server generates a password for a given student ID number ***idNum***. The ID-password encryption algorithm is as follows: For each five-digit decimal integer (the student ID number, i.e., ***idNum***), (1) Add a certain digit value, ***d***, (like 0, 1, .., 9) to each digit, divide the sum by 10 and use the remainder to replace the digit; (2) Swap 1st digit with 5th digit (the last digit); (3) Swap 2nd digit with 4th digit. (The middle digit, i.e., the 3rd one, stays in the same position).

Input: the first line gives an integer ***k*** which is the number of test cases and is followed by ***k*** lines of inputs, each of which (for a test case) contains two numbers, ***idNum***, ***d*** (as defined above) separated with a single white space.

- Example:

```
3
01234  5
99999  5
99996  6
```

Output: ***k*** lines of output, each of which is a five-digit number for the password.

- Example (the output for the above inputs):

```
98765
44444
25555
```

4. *DP_function*. Write a Java program that calculates function ***f***(***n***) for a given nonnegative integer, ***n***. Function ***f***(***n***) is defined as follows:

$f(0) = 0$
$f(1) = 1$
$f(2n) = f(n)$
$f(2n+1) = f(n) + f(n+1)$

Input: the first line gives an integer ***k*** which is the number of test cases and is followed by ***k*** lines of inputs, each of which (for a test case) contains the ***n*** defined above.

- Example:

```
5
1
3
123
1234
246
```

Output: ***k*** lines of output, each of which gives the value for ***f***(***n***).

- Example (the output for the above inputs):

```
1
2
14
61
14
```

5. *LSL*. Given a string (of at most of 256 characters), you are to write a Java program that outputs the longest substring without repeating characters. For example, the longest substrings for "ABCDDEFG" are "ABCD" and "DEFG", both of which have the length of 4.

Input: the first line gives an integer ***k*** which is the number of test cases and is followed by ***k*** lines of inputs, each of which (for a test case) contains a string.

- Example:

```
4
ABCDDEFG
ABCDABCDFEGE
IamAStudentOfSSU
IamaStudentOfSSU
```

Output: ***k*** lines of output, each of which gives the length for the inputted string.

- Example (the output for the above inputs):

```
4
7
10
8
```