


Attention

PC^2 displays ASCII value 0 as a box, and as a result this may cause judging errors, for example: an array of characters is auto filled with the ASCII value 0 which may display as ‘ ‘ in your output but appears as a box ‘’ on the judging machine.

Cy-Woods Programming Packet

February 2009

Student Packet

Problem Listing

2.1 James VS Zombies

2.2 The Price is Right?

2.3 The Key Master

2.4 Secret Password

2.5 Positional Cipher

2.6 Double Trouble

5.1 Kitty Talk

5.2 Riddle Writer

5.3 The Simple Birthday Problem

5.4 Math Teachers...

5.5 Transposition Cipher

5.6 EBG13 PVCURE!

9.1 Kitty Fencing

9.2 Windowpanes in the Well

9.3 Cat Burglar

9.4 A Web of Words

9.5 Doomsday

9.6 Wormhole Maze

Problem 2.1

James VS Zombies

General Statement: After a crazy party, James wakes up in the middle of an abandoned town. Before he can discover where he is, he is suddenly attacked by a horde of zombies! Luckily for James, he finds a conveniently placed shotgun next to him. His excitement dies quickly when he realizes that it only carries ammo that can defeat green zombies! James is only human, and can only take so much, so if more than 3 zombies (that aren't green) attack him, he's done for. Help James find out if he can survive this ambush.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain 10 characters, separated by a space. Each letter represents a different colored zombie, Green (G), Red (R), and Purple (P). All letters will be capitalized.

Name of Data File : pr21.dat

Name of Program : pr21.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should consist of a single sentence, stating whether or not James will survive and the amount of zombies he defeated. See the sample output for the two possible output formats. All letters will be capitalized.

Sample Input :

2

G G G G G P R G G R

R G R G G R G R G P

Sample Output:

JAMES MADE IT OUT ALIVE, DEFEATING 7 ZOMBIES ALONG THE WAY!

JAMES IS DONE FOR, BUT HE TOOK OUT 4 ZOMBIES WITH HIM!

Problem 2.2 The Price is Right?

General Statement: You have just been selected to be the next contestant on the Price is Right! Luckily for you, you manage to stumble across the retail prices of all the items in the showcases. If your opponent does not bid over the retail price, your bid will be \$1 more than his bid. In the case that your opponent bids over the retail price, all you need to do is bid \$1.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain two integers separated by a single space. The first will represent your opponent's bid and the second is the item's retail price.

Name of Data File : `pr22.dat`

Name of Program : `pr22.java`

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain your bid in the following format: `$number` as shown below.

Assumptions: Your opponent's bid will be at least \$1 above or below the actual retail price. All bids and prices will be positive amounts less than \$1,000,000.

Sample Input:

```
3
340 500
600 495
1336 2995
```

Sample Output:

```
$341
$1
$1337
```

Problem 2.3 The Key Master

General Statement: The Key Master uses an addition system to keep track of which keys unlock which rooms. Unfortunately, he has attained so many keys that he can't remember how to figure out his special addition! Write a program to help the Key Master figure out his system, which utilizes the following process:

Find the sum of two numbers.

Replace each digit of the number with its corresponding letter, with 0 representing A and 9 representing J. The portion of the alphabet used (A-J) corresponds, in order, with the digits 0-9.

Output the new series of letters created.

Input: The first line in the data file is an integer, n , that represents the number of data sets to follow ($1 < n < 100$). Each data set will contain two numbers to be added together to determine the code for the key.

Name of Data File : `pr23.dat`

Name of Program : `pr23.java`

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain the name of the door that the numbered key unlocks. All letters will be capitalized.

Assumptions: Only two numbers will be added together. The only letters used will be A – J. The numbers on the keys will never be decimals.

Sample Input:

```
4
99 1
9050 405
12 132
1 1
```

Sample Output:

```
BAA
JEFF
BEE
C
```

Problem 2.4 Secret Password

General Statement: Mr. A has a locked safe where he keeps his UIL materials hidden from all the jealous people who don't go to Cy-Woods. Since these materials are so highly desired, he changes the password on his lock once a week. Mr. A uses a password creator for his safe, but it has been malfunctioning. Mr. A wants his password to fit certain parameters, but recently the numbers it has been printing out don't always fit his parameters. This is where you come in. You are to write a program that takes in one of Mr. A's generator's passwords and outputs whether or not it is valid. For a password to be valid, it must fit all of the following parameters:

The password must be at least 6 digits.

The password must be an odd number.

The password must have a three in it.

The password must be divisible by 7.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set contains a single integer representing the number generated by the password generator.

Name of Data File : `pr24.dat`

Name of Program : `pr24.java`

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain either the word "VALID" or "INVALID".

Assumptions: Each input value will be an integer between `Integer.MIN_VALUE` and `Integer.MAX_VALUE`, inclusive.

Sample Input:

```
3
3079692
394
823543
```

Sample Output:

```
INVALID
INVALID
VALID
```

Problem 2.5

Positional Cipher

General Statement: A new cipher, called a positional cipher, is making it hard to decode secret messages Kevin is sending to Bradee. However, you, being the genius you are, discovered that the key string is "CYWOODSRULEZ" and each encrypted message simply describes the character's position in the string, subsequently removing each used character from the string. Use the key and your newly found rules to decode their messages.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain an encrypted string consisting of numbers 1-9, with a maximum length of 9.

Name of Data File : `pr25.dat`

Name of Program : `pr25.java`

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain a decrypted message.

Sample Input:

```
3
311
816
215
```

Sample Output:

```
WCY
RCS
YCS
```


Problem 2.6

Double Trouble

General Statement: All of this cipher business is finally getting to Bradee's head! All the messages he's trying to send to help sort out this packet are getting jumbled! Double occurrences of letters are being replaced in his messages with an asterisk ("*"). You need to pull those out and make the message make some sense.

An asterisk ("*") will follow some character in one of Bradee's messages. The asterisk denotes that a copy of the previous letter should occupy the asterisk's position in the sentence. Non-asterisk characters should be left alone.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set consists of a sentence of 1-100 words separated by spaces.

Name of Data File : pr26.dat

Name of Program : pr26.java

Output: Output the corrected message.

Sample Input :

4

THI** PA*SS* WO*RD
W*ORD* LI*KE THIS*
MO*NKE*Y B*ANANA PHON*ES
CYWOOD PROGRAM PACKET

Sample Output:

THIII PAASSS WOORD
WWORDD LIIKE THISS
MOONKEEY BBANANA PHONNES
CYWOOD PROGRAM PACKET

Problem 5.1 Kitty Talk

General Statement: Kevin is trying to talk to his cats, and he thinks he finally found out the mystery of the cat language. You need to write a program for him that reads in a line of human speech and outputs the line translated into Kevin's current theory of kitty language. The rules for kitty language are as follows:

Every character in the word turns into a type of meow. The letters are all coded to a different number, which determines how it is translated into kitty language. This translation consists of A = 1, B = 2, C = 3, etc. The base meow, representing the letter A, is "MEW". B is "MEEW", C is "MEEEW", etc, with the number of "E" s representing the letter's numeric value. The representation changes after the letter's numeric value increases beyond 10. If the number of "E"s exceeds 10, you may replace 10 "E"s with a "~" at the end of the meow. For example, 'K', which has a value of 11, should be printed as "MEW~". In addition, if the number of "E"s exceeds 20, an "O" may similarly be added before the "W" in replacement of 10 more "E"s. This is seen in the letter W, which has a value of 23, and is represented as "MEEEOOW~".

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set consists of a word to be translated from human speech to kitty speech. All letters will be capitalized.

Name of Data File : pr51.dat

Name of Program : pr51.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain the kitty language representation of the input word. All letters will be capitalized.

Sample Input:

```
3
HELLO
CAT
WORLD
```

Sample Output:

```
MEEEEEEEEW MEEEEEW MEEW~ MEEW~ MEEEEEW~
MEEW MEW MEEEEEEEEEEW~
MEEEOOW~ MEEEEEW~ MEEEEEEEEEW~ MEEW~ MEEEEW
```

Problem 5.2

Riddle Writer

General Statement: Alex Ameen is trying to find a way to make his data look more confusing. He has gotten increasingly lazy, and has finally decided to write a program to do this for him. Unfortunately Alex has proven too lazy even for this, so you will need to write the program for him! The riddle works by counting the number of times each character in a string is repeated, then printing out the number of times, followed by the character. For example, "AAAAA" consists of 5 As, so the riddle-fied version would be "5A". "AABBCCDD" consists of 2 As, 2 Bs, 2 Cs, and 2 Ds so the riddle-fied version would be "2A2B2C2D".

Input: The first line of data will contain an integer with the number of data sets. Each data set will contain the string of text that needs to be riddle-fied. All letters will be capitalized.

Name of Data File : pr52.dat

Name of Program : pr52.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain a riddle formatted using the above rules. All letters will be capitalized.

Assumptions: Each string will consist of only alphanumeric characters.

Sample Input:

```
4
A
AACCC3333
117777770000111
1
```

Sample Output:

```
1A
2A3C43
21674031
11
```

Problem 5.3 The Simple Birthday Problem

General Statement: Mr. Wong, the mathematician/magician, has spent his life coming up with the formula for the Birthday Problem. With a little bribery, he gives you his life's work. The equation is as follows:

$$p(n) = 1 - e^{-\frac{n(n-1)}{2 \cdot 365}}$$

n = the number of people

e = Euler's number, to be approximated to 2.718281828459045

*Note: The formula reads "1 minus Euler's number **to the power of** negative...".

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain 1 integer, n .

Name of Data File : pr53.dat

Name of Program : pr53.java

Output: Output the correct probability given n number of people that at least two of them will share a birthday. Format the output to three decimal places.

Assumptions: $n \geq 1$ and $e = 2.718281828459045$.

Hint: Java's Math class has a static value for Euler's Number.

Sample Input:

```
3
2
10
23
```

Sample Output:

```
PROBABILITY IS: 0.274%
PROBABILITY IS: 11.599%
PROBABILITY IS: 50.000%
```

Problem 5.4

Math Teachers...

General Statement: Your Math teacher is not as smart as she would like you to believe and cannot add in binary. She needs you to write a program that will do the binary math for her; however, she's not quite sure what a binary number is, and may give you non-binary numbers.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each line contains two binary numbers.

Name of Data File : pr54.dat

Name of Program : pr54.java

Output: Your program should produce n lines of output, n representing the number of data sets. Print out first binary number + second binary number = sum of both binary numbers or "EPIC FAIL!!!" if one of the numbers given is not a binary number.

Assumptions: The two binary numbers in a single case will be the same length.

Sample Input:

```
5
0010 1100
1111 1111
0001 0001
1210 1111
1111111 1000001
```

Sample Output:

```
0010 + 1100 = 1110
1111 + 1111 = 11110
0001 + 0001 = 10
EPIC FAIL!!!
1111111 + 1000001 = 11000000
```

Problem 5.5 Transposition Cipher

General Statement: You have been tasked with encrypting your company's passwords. You decided to encrypt the passwords with a Transposition Cipher, which is set up as follows:

The password will be less than 10 words. Each word in the password has the same number of letters, and the length of each word is even. For example: THIS PASS WORD. The words are then positioned like this:

```
T I
H S
P S
A S
W R
O D
```

The words are then collapsed to left like so:

```
TI
HS
PS
AS
WR
OD
```

Then you read from the top of each column down resulting with the encrypted password of:

```
THPAWO ISSRD
```

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set contains no more than 10 words and no less than 2 words. All letters will be capitalized.

Name of Data File : pr55.dat

Name of Program : pr55.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain an encrypted password. All letters will be capitalized.

Sample Input :

4

```
THIS PASS WORD
WORD WILL LOOK LIKE THIS
MONKEY BANANA PHONES
ELEPHANT MANAMANA FUNINPUT
```

Sample Output:

```
THPAWO ISSRD
WOWILOLITH RDLLOKKEIS
MOBAPH NKNAON EYNAES
ELMAFU EPNANI HAMANP NTNAUT
```

Problem 5.6

EBG13 PVCURE!

General Statement: You are cruising through the internet trying to find more information about a top secret Area-51 document. You stumble upon the government's files, but you find them to be encrypted with the Rot13 Cipher. Just as you start decrypting their files, the system is alerted to your presence! It is up to you to write a program to decrypt the Rot13 Cipher before the internet tubes get clogged.

The Rot13 Cipher acts as follows. The alphabet is divided in half, and the first half is aligned with the second to create the translation as such:

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

For example "HELLO" becomes "URYYB". If any punctuation or numbers are present, leave them as is. All letters will be capitalized.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain 1 line.

Name of Data File : pr56.dat

Name of Program : pr56.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain a correctly decrypted message.

Assumptions: the input will only contain capital letters

(ABCDEFGHIJKLMNOPQRSTUVWXYZ), numbers (0123456789), and punctuation

(!"#\$%&'()*+,-./:;<=>?@[\\]^_`{|}~).

Sample Input:

```
3
EBG13 PVCURE!
NOPQRSTUVWXYZABCDEFGHIJKLM
PL-JBBQF 4 YVSR!
```

Sample Output:

```
ROT13 CIPHER!
ABCDEFGHIJKLMNOPQRSTUVWXYZ
CY-WOODS 4 LIFE!
```

Problem 9.1 Kitty Fencing

General Statement: Kevin is fencing off part of his future giant mountain ranch (in the future) into a rectangular kitty play area. He spent all his money on his giant ranch and his millions of kitties though, so he doesn't have very much money left for fencing. Fortunately, as Kevin's future mountain ranch is in the mountains, he may be able to use the sides of the mountains as edges of his kitty play area, and therefore won't need a fence on that edge.

In order to decide what dimensions his kitty play area should be in order for him to have the maximum possible kitty play area, Kevin wants a program written for him (he'd write it himself, but he's too busy playing with his kitties). Your program will need to take in the amount of money Kevin has, the cost of each yard of fencing, and the number of sides that will need to be fenced, then output the dimensions that the kitty play area should be.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain 3 integers. The first integer represents the money Kevin has to spend on fencing (\$0 to \$1,000,000), the second integer represents the cost of each yard of fencing (\$0 to \$300), and the third integer represents the number of sides that will need to be fenced (1 to 4).

Name of Data File : `pr91.dat`

Name of Program : `pr91.java`

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain the length and width (in yards) of the maximum possible kitty play area, formatted as seen below in the sample output. If it is impossible to determine a dimension (either or both dimensions could be indeterminable), output "UNKNOWN" for that dimension. Print the longer dimension first.

Sample Input:

```
4
5 1 4
14 2 3
2 4 2
1200 2 1
```

Sample Output:

```
1.25 x 1.25 YARDS
3.50 x 1.75 YARDS
0.25 x 0.25 YARDS
600.00 x UNKNOWN YARDS
```


Problem 9.2 Windowpanes in the Well

General Statement: Oh no! Matt Darby fell down the well! Fortunately, his sister, Lassie, went for help. Unfortunately, no one could retrieve him from the well. Matt was left with no decision but to make the well his permanent residence. All he needs now are some decorations! The first thing that Mr. Darby wants is a window. You feel so sorry for his genetics that you decide you will make one for him. Given the number of panes Matt wants, you are to output some blueprints for the window framing that will need to be constructed.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain a single integer, representing the number of panes the window frame is to have.

Name of Data File : `pr92.dat`

Name of Program : `pr92.java`

Output: Your code should output the schematics for the window frame, formatted as shown below. Each windowpane will be the same size (4x4). Include one line of space between each data set's output.

Assumptions: The number of desired windowpanes will always have a whole square root. Matt cannot request more than 600 window panes.

Sample Input:

```
2
4
9
```

Sample Output:

```
XXXXXXX
X  X  X
X  X  X
XXXXXXX
X  X  X
X  X  X
XXXXXXX

XXXXXXXXXX
X  X  X  X
X  X  X  X
XXXXXXXXXX
X  X  X  X
X  X  X  X
XXXXXXXXXX
X  X  X  X
X  X  X  X
XXXXXXXXXX
```

Problem 9.3

Cat Burglar

General Statement: Sir Fluffykins, Esq. is the world's most famous cat burglar. He, in an attempt to best all of his previous feats of burgling, decides to steal the Cat's-Eye emerald from the most prestigious jewelry museum in all of Hoboken, New Jersey. Everything goes perfectly according to plan until Sir Fluffykins, Esq. triggers the newest security measure installed to protect the emerald: a bomb! The only way to disarm the bomb and escape to burgle again is for Sir Fluffykins, Esq. to take the possible key values and try to disarm the bomb with every combination before the bomb explodes. Write a program to calculate if Sir Fluffykins, Esq. can determine the code from the initial values within the time limit.

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain 3 strings, each one being on a separate line: a string of possible key values, a string showing the actual key required to disarm the bomb, and an integer representing the number of seconds Sir Fluffykins has to disarm the bomb.

Name of Data File : pr93.dat

Name of Program : pr93.java

Output: If Sir Fluffykins can disarm the bomb in time, print out "THE DAY IS SAVED. SIR FLUFFYKINS LIVES TO BURGLE AGAIN!" Otherwise, print out "BOOM! NO MORE BURGLING FOR SIR FLUFFYKINS!"

Assumptions: Constructing the final code from the initial values requires 1 second per initial construction. For example, given the key values of 123 and a final key of 321, we construct 1, 12, 123, 13, 132, then 2, 21, 213, 23, 231, then 3, 31, 312, 32, 321, in that order, requiring 15 seconds (1 second per construction). If the key value string IS the disarm code, it requires 0 seconds.

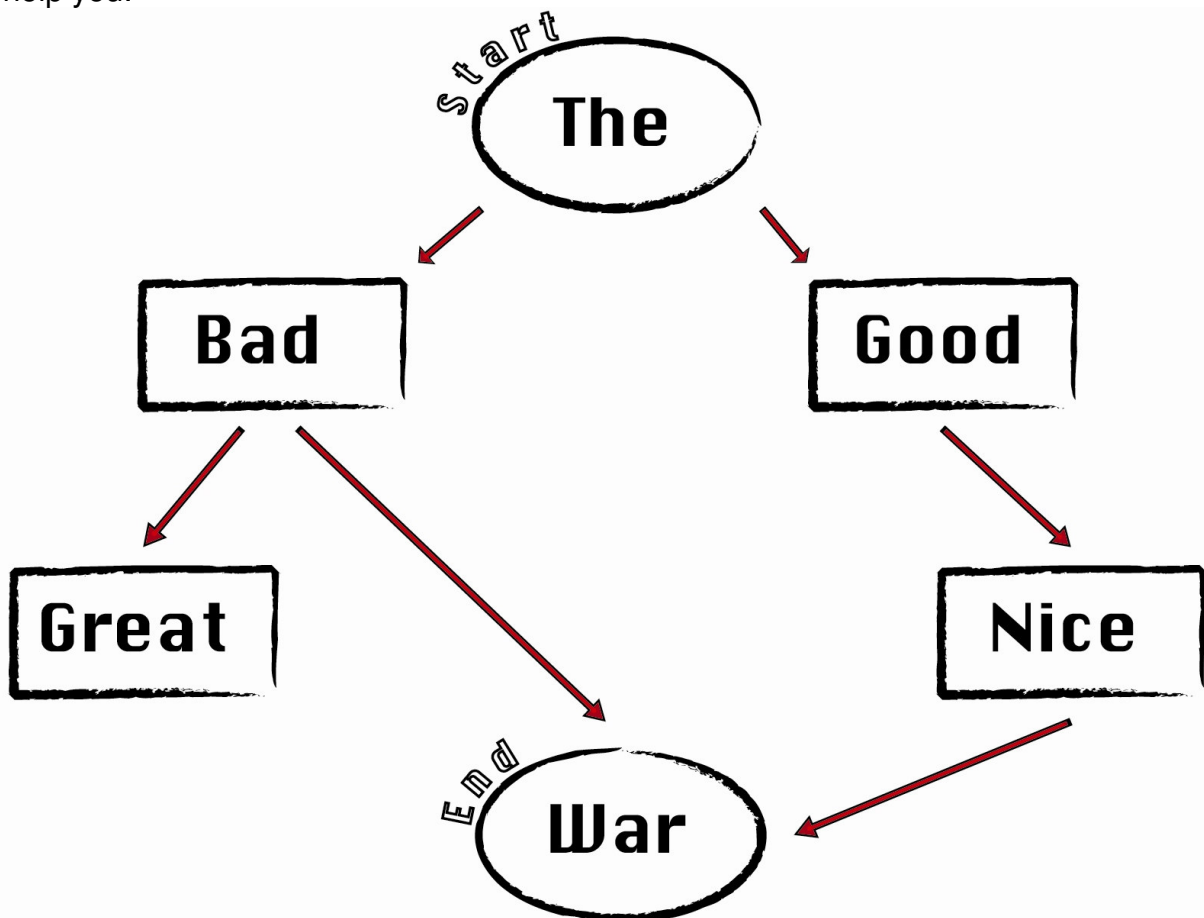
Sample Input :

```
3
123
321
3
123
231
14
111111
111111
1
```

Sample Output:

```
BOOM! NO MORE BURGLING FOR SIR FLUFFYKINS!
THE DAY IS SAVED. SIR FLUFFYKINS LIVES TO BURGLE AGAIN!
THE DAY IS SAVED. SIR FLUFFYKINS LIVES TO BURGLE AGAIN!
```

General Statement: Bradee thinks himself the "most cleverest" (his words) of all Computer Science students, so he encodes all of his secret messages in the form of a web of interconnected words. The secret message is derived from the shortest path of words from the starting word to the end word (both given). Write a program to print out the secret message! Use the diagram representing the first data set in the sample input to help you.



Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain, on the first line, the number of paths to be given, n , followed by the starting and ending words separated by a space. The next n lines will contain a path per line, consisting of the first word and the word it connects to in the web. A path consists of ONLY two words.

Name of Data File: pr94.dat

Name of Program: pr94.java

Output: For each data set, output the secret message, aka the shortest connecting path of words that connect the starting and ending words.

Assumptions: There will be no more than 100 paths given. The starting and ending words will not be the same word. There will always be a connecting path, and there will be no ties for the shortest path.

Sample Input :

(page 2 of 2)

```
2
6 THE WAR
THE BAD
BAD WAR
THE GOOD
GOOD NICE
NICE WAR
BAD GREAT
5 THEENTS THEWAR
THEENTS AREGOING
AREGOING TOWIN
TOWIN THEWAR
THEENTS WHATCATS
WHATCATS AREGOING
```

Sample Output:

```
THE BAD WAR
THEENTS AREGOING TOWIN THEWAR
```

Problem 9.5 Doomsday

General Statement: The Doomsday is a vital component in the Doomsday Rule, which is able to determine the exact day of the week of any day in any year. The rule takes advantage of the fact that within any calendar year, the days of 4/4, 6/6, 8/8, 10/10, 12/12, and the last day of February always occur on the same day of the week. The aforementioned dates' day of the week parallel the year's doomsday.

The algorithm to find the Doomsday is as follows:

where y = the last 2 digits of the given year (i.e. in 1966, $y = 66$)

$$\left(\left\lfloor \frac{y}{12} \right\rfloor + y \bmod 12 + \left\lfloor \frac{y \bmod 12}{4} \right\rfloor \right) \bmod 7 + \text{anchor} = \text{Doomsday}$$

The Anchor day is as follows:

Century	Anchor day
1800-1899	Friday
1900-1999	Wednesday
2000-2099	Tuesday
2100-2199	Sunday

This algorithm involves treating days of the week like numbers mod 7 so:

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
0	1	2	3	4	5	6

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain a four digit integer representing the year.

Name of Data File : pr95.dat

Name of Program : pr95.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain the year's doomsday in the following format:
DOOMSDAY LIES ON : DAY-OF-WEEK

Assumptions: $1800 \leq y \leq 2199$

Sample Input:

```
4
1966
2005
2009
1950
```

Sample Output:

```
DOOMSDAY LIES ON : MONDAY
DOOMSDAY LIES ON : MONDAY
DOOMSDAY LIES ON : SATURDAY
DOOMSDAY LIES ON : TUESDAY
```

General Statement: Matt Darby is traveling through space attempting to hack the universe. Using code he found on Wikipedia, Darby calculates the location of the center of the universe. Upon arriving in the center of the universe, a black hole sucks him in to an alternate dimension. After a rough trip through the black hole, Darby discovers a strange new reality. Wormholes connect different paths throughout this strange new world, and there is only one way to escape. Since Darby's Wiki code is useless at this point, he turns to you for help. Help Darby solve the puzzle of the wormholes and escape the center of the universe!

Input: The first line in the data file is an integer that represents the number of data sets to follow. Each data set will contain two integers, r and c , and a wormhole maze, which consists of one or more lines of various characters. The two integers in the data set specify the size of the maze. r , the first integer, represents the height of the wormhole maze, and c , the second integer, represents the width. The maximum size of the matrix is 60x60 and the minimum size is 1x3 (rows x columns).

The following are the only symbols that will be used:

- @ - starting position
- ! - exit
- # - wall
- . - space
- Aa - wormholes

Inside the mazes are pairs of wormholes. Each pair of wormholes has an entry wormhole and an exit wormhole, meaning the wormholes are one-way paths. Uppercase letters represent the entry wormholes, and lowercase letters represent the exit wormholes. Matt cannot pass through exit wormholes, he may only move out of them.

Name of Data File : pr96.dat

Name of Program : pr96.java

Output: Your program should produce n lines of output, n representing the number of data sets. Each output line should contain one sentence stating the shortest number of steps it takes to go through the wormhole maze, like so: "ESCAPED THE WORMHOLES IN x STEPS!", where x is the shortest number of steps.

Assumptions: The starting position will always be spot (0,0) of the maze. Matt will always be able to escape. For every entry wormhole, there will be an exit wormhole.

Moving/warping from an entry wormhole to an exit wormhole does not count as a step. It will always take more than 1 step to escape the maze. Matt cannot move diagonally.

Sample Input:

```
3
1 7
@.A#a.!\n5 5\n@####\n....#\n#..##\nA.###\n##a.!\n11 10\n@#####\n.#a#b#d.##\n.#.#.#..##\n.#.#.#..##\n.#.#.#E.e#\n.#.#C#f.F#\n.#.###G.g#\n.#.#c#h.H#\n.#.#.#I.i#\n.#.#.#j.J#\nA#B#D#K.k!
```

Sample Output:

```
ESCAPED THE WORMHOLES IN 4 STEPS!\nESCAPED THE WORMHOLES IN 7 STEPS!\nESCAPED THE WORMHOLES IN 36 STEPS!
```