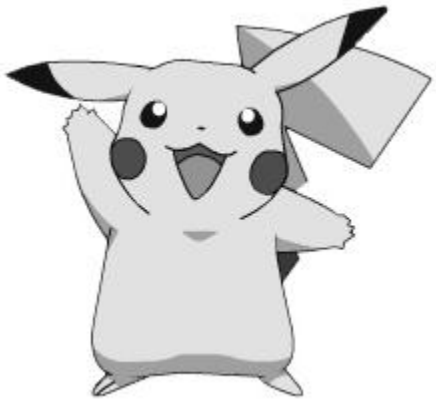
CRHS UIL Computer Science Contest 2011



While you're waiting for the contest to start...some PokéFacts!

- Magikarp may be useless until it evolves, but base-stat wise, it is only the 8th weakest Pokémon. The weakest is actually Sunkern.
- Gastly and Haunter are tied as the lightest Pokémon, while Groudon is the heaviest.
- Notice the hidden numbers in Artic**uno**, Zap**dos**, and Mol**tres**, the legendary bird Pokémon. Mind == Blown
- Hitmonchan was named after Jackie Chan, while Hitmonlee was named after Bruce Lee.
- There's an unused feature in Gold and Silver where you can name your mom. It can be enabled via hacking and is fully functional in-game.

In the anime, every Officer Jenny has a unique symbol on her hat that indicated which city she came from.

PokédexPhailure

Input File: pokedex.dat

Problem Description: While Ash was browsing the InterWebs on his Pokédex looking for Pokémon walkthroughs, Ash's Pokédex was infected with the Pokérus (Pokémon Virus). As a result, the Pokédex internal Pokémon database is scrambled. When the Pokédex accesses stored Pokémon in its memory, it incorrectly interprets the stored codes, thus displaying the incorrect Pokémon. A Gengar stored in memory may be displayed as an Abra when the Pokédex retrieves the data Because Ash is too busy trying to train his Pokémon on double EXP, Pikachu is left with the task of repairing the Pokédex. Help Pikachu rewrite the Pokédex database.

Input: The first line of input is an integer A. The next A lines will be data sets of the correct database. Each entry in the database will be formatted as follows:
PokémonCode+" "+PokémonName.

For example:
`B00 BULBASAUR`
When the Pokédex interprets the code "`B00`" it will display
BULBASAUR to the screen.

The next line of input after the correct database is an integer B and the next B lines will be data sets of the scrambled database formatted in the same way as the correct database. On the line after the scrambled database will be an integer C. The next C lines will be Pokémon displayed to the screen when the Pokédex using the scrambled database to interpret the codes. Each Pokémon will be on one line.

Output: Output the actual Pokémon that Ash saw. Output should be capitalized. In the sample below, the first Pokémon displayed by the Pokédex is "`PIDGEY`". The Pokédex displayed "`PIDGEY`" because in the scrambled database, the line "`C10 PIDGEY`" indicates that the Pokédex should interpret the code "`C10`" as a "`PIDGEY`". However, looking at the correct database, the code "`C10`" should represent a "`BUTTERFREE`". Because the actual Pokémon should have been a BUTTERFREE, output BUTTERFREE.

Assumptions: All Pokémon codes are 3 characters long and may contain capital letters and integers. All Pokémon names will be capitalized. The correct database and the scrambled database will contain the same Pokémon. The list of Pokémon displayed on the Pokédex will always be in the databases. In each database, codes and Pokémon will not be repeated.

Sample Input:
```
6
B00 BULBASAUR
E00 CHARMANDER
H00 SQUIRTLE
G10 PIDGEY
C10 BUTTERFREE
C20 PIKACHU
6
C20 BULBASAUR
G10 CHARMANDER
B00 SQUIRTLE
C10 PIDGEY
H00 BUTTERFREE
E00 PIKACHU
2
PIDGEY
CHARMANDER
```

Sample Output:
```
BUTTEFREE
PIDGEY
```

Super Effective

Input File:super.dat

Problem Description: Ash is battling inside a Pokémon tournament using only a single Pokémon: Pikachu. Every time Pikachu gets a One-Hit-KO, the announcer screams “Wow!! That was SUPER EFFECTIVE!!!” even when the skill is not super effective. However, after a while the announcer gets tired of screaming the same words into the microphone and resorts to printing the words “Wow!! That was SUPER EFFECTIVE!!!” to the big screen in the stadium. The screen is hooked up to a sensor which detects the stats of each battling Pokémon and analyzes it in order to determine if “Wow!! That was SUPER EFFECTIVE!!!” is written to the screen. Write the code for the sensor.

Input: The first integer N indicates that N data sets will follow. Each data set occupies one line. Each line of input will contain 6 integers A, B, C, D, E, F separated by spaces.
A - represents Pikachu’s level.
B - represents Pikachu’s attack rating.
C - represents base attack of the skill Pikachu uses.
D - represents a random value
E - represents opponent’s defense rating.
F - represents opponent’s hit points.
The formula to determine damage dealt is:
Damage = ((((((A × 2 ÷ 5) + 2) × B × C ÷ 50) ÷ E) + 2) × D ÷ 100)

Output: Using the damage formula, if the damage done is greater than or equal to F (the opponent’s hit points) then a One-Hit-KO will occur. If Pikachu does land a One-Hit-KO, print out “`Wow!! That was SUPER EFFECTIVE!!!`” to the screen. If Pikachu does not do a One-Hit-KO, print “`...`” to the screen.

Assumptions: Input for B, C, E will be between 1 and 500 inclusive. Input for A, D will be between 1 and 100 inclusive. All damage should be rounded down to the nearest integer (ex. If Pikachu does 93.8 damage, it should be rounded down to 93 damage).

Sample Input:

```
4
100 25 99 2 6 100
100 180 256 94 83 400
100 60 123 93 28 200
8 16 36 50 1 27
```

Sample Output:

```
...
Wow!! That was SUPER EFFECTIVE!!!
Wow!! That was SUPER EFFECTIVE!!!
Wow!! That was SUPER EFFECTIVE!!!
```

PokéFinder

Input File: pokefinder.dat

Problem Description: In every trainer's futile quest to "catch 'em all", there is an inordinate amount of wandering around to find Pokémon. Automate this task by building a program that can search a grid of characters to find the names of several Pokémon. Only search horizontally and vertically, no diagonals.

Input: The first line of input contains an integer, n, indicating the number of rows. The second line contains an integer indicating the number of columns. The next three lines contain the Pokémon that you are searching for. The remaining n lines will be the contents of the word grid. Each character is separated by a space.

Output: Print out the position of the first letter of the name and the direction to go. Row and column numbers start at 1. Rows count down from the top and columns count across from the left. If the Pokémon you are searching for is not in the grid, display "NAME_OF_POKEMON `could not be found`".

Sample Input:

```
10
10
PIKACHU
SNORLAX
BUBBLESORT
Q K E B H U C U Z V
P V J U E I Y H R Z
Z L W B A R T C A R
Y M E B I H A A H R
D F V L E V E K C I
U I F E G Z Q I V J
M U D S E Q F P Q Y
Q T H O Q R H T D F
Q T N R W U Z G B K
E L L T U J T Z V Q
```

Sample Output:

```
PIKACHU is found going Up from row 7, column 8.

SNORLAX could not be found.

BUBBLESORT is found going Down from row 1, column 4.
```

Pokémon Daycare

Input File: daycare.dat

Problem Description: Here at the hatchery, all the little Pokémon eggs need constant attention in their incubators. Subsequently, some number of nurses regularly patrols the row of an equal number of cribs, opening the crib to look inside. Each nurse has a different rotation, devised so that she checks each crib that has an ID# (1- num) that her ID# (1- num) is a multiple of. Note, however, that these nurses constantly leave the lids for the cribs open; whenever a nurse comes across a closed crib, she opens it, and each time she comes an open crib, she closes it. Whenever a nurse checks a crib, she makes a tally mark under its number.Find the crib(s) that have the greatest amount of tally marks at the end of the day. But wait, there's more! Every crib left open after the last inspection is left open all night, and the egg becomes rotten. Find the crib(s) that have been left open so that the owners may be informed. Each crib starts off closed before the nurses arrive.

Input: The first number is number of data sets. Each data set contains a single integer representing the number of nurses and cribs.

Output: The crib(s) that were visited the most times, and the crib(s) that were left open.

Sample Input:

```
2
50
10
```

Sample Output:

```
The crib(s) that were visited the most are 48
Oh No! Crib(s) 1, 4, 9, 16, 25, 36, 49 were left open! What a shame.

The crib(s) that were visited the most are 6, 8, 10
Oh No! Crib(s) 1, 4, 9 were left open! What a shame.
```

PokéSort

Input File: pokesort.dat

Problem Description: Create a program that takes in the information about each Pokémon and sorts them in the order of their combat ability. A Pokémon's combat ability is determined by its attack and defense points as well as its speed and agility. Attack and defense points are three times more important than speed and agility points.

Input: Your program will take in a number indicating the number of Pokémon, and the names of each Pokémon followed by four numbers, indicating its attack, defense, speed, and agility points respectively. Each new line contains the information of a different Pokémon.

Output: Your program will print out the names of the Pokémon in order of their combat ability.

Assumptions: The names will not contain spaces.

Sample Input:

```
5
Charz 2 6 5 7
Pikachu 5 7 4 6
Bubblesort 6 7 3 5
Mudkit 9 8 23 3
Azian 1 2 99 99
```

Sample Output:

```
Azian
Mudkit
Bubblesort
Pikachu
Charz
```

Pokémon Calls

Input File: calls.dat

Problem Description: Pokémon make sounds based off the first few letters of their name. Common examples include Pikachu (Pika) and Charizard (Char). Write a program that, when the name of a Pokémon is entered, returns the sound it will make (the first 4 letters of the name). If a Pokémon has less than 4 letters in its name, its call is its entire name

Input: You will receive an integer for how many calls you need to determine, followed by the Pokémon names, one per line.

Output: Print out the calls one by one, one per line.

Sample Input:

```
3
Pikachu
Charizard
Muk
```

Sample Output:

```
Pika
Char
Muk
```

Catch Rate

Input File: catch.dat

Problem Description: Pokémon has a very precise catching algorithm. Whether or not a Pokémon is caught depends on :

* its 8 bit catch rate (on a scale from 0 to 255)
* the type of ball it is caught with
* its total HP and its current HP (on a scale that is predetermined)
* a status ailment, like sleep, poison, paralysis, or burn.

The formula is this:

$$a = \frac{(3HP_{max} - 2HP_{current})(rate)(ball_{multiplier})}{(3HP_{max})} + status_{multiplier}$$

Input:

* The first is an integer indicating how many Pokémon scenarios should be tested
* That Pokémon's catch rate (0 to 255)
* A string indicating the type of ball (will be one of these: "`Pokeball`", "`Great Ball`", "`Ultra Ball`", "`Master Ball`")
* Current HP
* Total HP
* A string indicating a status condition (`Paralysis`, `Sleep`, `Poison`, `Burn`) or none (`none`).
* The Pokémon's level.

A status multiplier is 100 for sleep and poison, 50 for everything else.
A pokéball's multiplier is 2x for a Great Ball, 4x for an ultra ball, and a master ball always catches the Pokémon no matter what.

Output: If the value calculated is greater than its level times 3, the Pokémon is caught. Print out "`GOTCHA!`" if it's caught. Print out "`OH NO!`" if it can't be caught.

Assumptions: none

Sample Input:

```
2
125
Great Ball
23
100
None
5
25
Ultra Ball
100
125
Paralysis
100
```

Sample Output:

```
GOTCHA!
OH NO!
```

Level Up

Input File: levelup.dat

Problem Description: When Pokémon level up, their stats improve as such:

Hit Points	+10%
Attack	+10%
Defense	+10%
Required XP for next level	+50%

Each Pokémon starts out with 30 Hit Points, 20 Attack, 12 Defense, and 50 next-level XP.

Compute a Pokémon's stats after it receives a given amount of XP points.

Input: The first line of input is the number of data sets to follow. Each data set consists of a single XP point value.

Output: The Pokémon's new stats in the following format. All values will be integer. The last value to level up is not dependent of its current XP; only the next level.

> Your Pokémon is now at level _, with _XP left. He has _HP, _A, _D, and requires _XP to level up.

Assumptions: After each level gain, truncate the XP points (round down).

Sample Input:

```
3
200
50
5
```

Sample Output:

```
Your Pokemon is now level 3, with 75XP left. He has 36HP, 24A, 14D, and requires
112XP to level up.
Your Pokemon is now level 2, with 0XP left. He has 33HP, 22A, 13D, and requires
75XP to level up.
Your Pokemon is now level 1, with 5XP left. He has 30HP, 20A, 10D, and requires
50XP to level up.
```

Magikarp

Input File: magikarp.dat

Problem Description: Being a trainer who is not the main character, your whole team is composed of magikarp. They are of different levels, but all of them only knows splash. Given a list of the magikarps' amount of HP, find how many turns your team can last before the main character steals your money. (Aka, all are defeated).

Input: First line is an integer of the number of rounds. For each round, the first integer is the number of magikarp you have followed by the HPs of each one.

Output: For each round, print out the number of total hits your team takes before you lose.

Assumptions: You have no more than 6 magikarp in each round. You do not switch magikarp in each round. Each hit imposed on your magikarp removes 20HP. All attacks will hit.

Sample Input:

```
3
5
240
350
459
562
100
3
44
250
45
1
1000
```

Sample Output:

```
87
19
50
```

Meowth

Input File: meowth.dat

Problem Description: Meowth is learning how to speak. He's still in the learning stages, where he only has a short vocabulary of words, and he still can only say "meowth." The speech trainer can still understand him by telling how long he says the word "meowth". The length of the word "meowth" determines how long the English word is minus 5. Meowth is learning how to order his words, so the speech trainer reprimands him if his grammar is incorrect. Analyze Meowth's words to see if he's using proper word order. If he says a word that is not in his vocabulary, he fails the test as well.

Input: The first number will be the length of Meowth's vocabulary. The next number will be the number of phrases you need to check. Each set of phrases will consist of the English phrase Meowth is translating, and then Meowth's response.

Output: If meowth said the right thing, print out "`Good Job, Meowth!`". If he said the wrong thing, print out "`That's the wrong order!`" If he says something not in the vocabulary, print out "`What did you say?`"

Assumptions: None of the words in Meowth's vocabulary will have the same amount of letters. The word he says ("`meowth`") will always contain more than 6 letters; if it does not, Meowth said something that wasn't in his vocabulary.

Sample Input:

```
5
Hey
I
am
doing
fine
2
Hey I am doing fine
Meeeowth meowth meoowth meeeoooowth meeeeowth
Hey I am doing fine
Meeeeowth meeeoooowth meoowth meowth meeeowth
Hey I am doing fine
Meeeeeeeeeeeeeeeowth meeeoooowth meoowth meowth meeeowth
```

Sample Output:

```
Good Job, Meowth!
That's the wrong order!
What did you say?
```

Oak's Quest

Input File: oak.dat

Problem Description: Professor Oak is a Pokémon researcher. He needs to find the next town to do his studies. In addition to studying Pokémon, Professor Oak also likes to make friends with the ladies of the towns. Unfortunately, all of the single women in the country are moms and Professor Oak does not like dealing with children. He has a plan, however. If the child is old enough (10 or older), he can send them on quests to look for Pokémon. Help Oak find the town with the most single mothers that he can mingle with.

Input: First line is the number of towns to analyze. For each town, the first line will be the town name. The second line will be the number of households. For each household, the first number is the number of people. For each person, the first line is a string telling you if he/she is a mother, father, or children. (Hint: if there is more than one mother, Oak does not have a chance, so don't count them!) The second line is their age.

Output: Print out the town name that Oak should set up his lab and the number of single moms with all their children at least 10 years old.

Assumptions: All the single women have at least one child. There is only a max of two parents. One town will always be the best one.

Sample Input:

```
3
LITTLEROOT
4
3
MOTHER
23
CHILD
14
CHILD
13
3
MOTHER
27
FATHER
26
CHILD
11
2
FATHER
29
CHILD
13
2
MOTHER
26
CHILD
9
PALLET
2
2
MOTHER
28
CHILD
18
2
```

```
MOTHER
26
CHILD
14
LAVENDER
5
2
MOTHER
19
CHILD
6
1
FATHER
34
2
MOTHER
29
CHILD
17
2
MOTHER
25
CHILD
14
3
MOTHER
31
CHILD
11
CHILD
11
```

Sample Output:

```
LAVENDER 3
```

Route 101

Input File: route.dat

Problem Description: You are a trainer attempting to cross a route to a Pokémon center across the road. Your Pokémon are weak after a day of battling, and they can only defeat a few more trainers successfully. You look across the road and see a few trainers ahead. To give your Pokémon rest, you do your best to traverse the road avoiding as many trainers as possible. If you battle more trainers than you have Pokémon, you will "white out" and be returned to your starting point. Can you make it to the end avoiding as many trainers as possible?

You will receive a grid of characters showing the route ahead of you. The field will always be eight characters wide, but it is of variable length. Grass will be denoted as asterisks (*) and trainers will be denoted as one of the following:

> (greater than)	is a trainer facing right.
< (less than)	is a trainer facing left.
v (letter "v")	is a trainer facing downward.
^ (a carat)	is a trainer facing upward.

Your starting position will be denoted as a capital 'S'. You can assume you will not be in any trainer's line of sight in this position. Your destination will be denoted as a capital 'D', and the same assumption can be made.

Every trainer has a line of sight that extends until it reaches the edge of the map, or is interrupted by another trainer. The line of sight will not be shown in the map. If you cross this line of sight, you will have to battle this trainer, using up some of your Pokémon's strength. The lines of sight can intersect, however you may assume two lines of sight will never be collinear (i.e. two trainers will never face each other unless there is another trainer in the way). Once a trainer has been battled, his line of sight is crossable again. However, the trainer will still be there to block your path and other trainer's lines of sight (i.e. trainers will stay in the same place).

Input:

The first integer will be a number of data sets to consider. ≠ 0
The second integer will be the number of trainers your Pokémon are prepared to battle. ≠ 0
The third integer will be the length of the route ahead. ≠ 0
The map will be given to you afterward, and it will not exceed line length of the third integer.

Output:

If the destination is reachable, print out the least possible number of trainers you battled. This should never be larger than the second input integer.
If the destination is unreachable, print out "PLAYER WHITED OUT!"

Sample Input:

```
2
3
5
****S***
>*******
******^*
**^****<
******D*
1
2
S*v*****
*****^*D
```

Sample Output:

```
2
PLAYER WHITED OUT!
```

Sprites

Input File: sprites.dat

Problem Description: You and a friend have decided to make a Pokémon clone based in ASCII. However you wish to use the original GameBoy sprites by converting them into ASCII. In order to convert the sprites you must first obtain them. Luckily your friend has already written a program in C that copies sprites from memory and stores them in a file. However, your partner has forgotten to decode the raw byte sprite data into an ASCII format. You must write a program to decode and draw the sprite in ASCII.

A GameBoy sprite is 8x8 pixels represented by 8 bytes. Every bit in each byte signifies a pixel in a sprite. If a bit is high a pixel should be drawn The pixel coordinate system is the same as the default swing coordinate system; (0,0) being the upper most, left most pixel, (7,7) being the bottom most right most pixel.

For example:

The 0 bit in the first of the eight bytes represents the pixel at (7,7).
The 3 bit of the fourth byte signifies the pixel at (3,5).

You will convert the 8x8 sprite into an 8x8 character array with each pixel/bit corresponding to a character. If a bit is high and a pixel should be drawn, then use the character '@'to represent the pixel in ASCII. Otherwise, use the underscore character '_'.

In addition to the sprite there is an 8x8 pixel mask that is to be applied to each sprite. The pixel mask data is represented in the same 8 byte manner as the sprite data.

Input: The file will contain 16 bytes on a single line. The first 8 bytes will represent the pixel data. The remaining 8 bytes will represent the mask to apply to the pixel data.

A mask will erase a pixel row off if it is 0 and leave the pixel row unchanged if it is -1.

Sample Input:

```
124 -58 -58 -2 -58 -58 -58 124 -1 -1 -1 -1 -1 -1 -1 0
```

Sample Output:

```
_@@@@___
@@___@@_
@@___@@_
@@@@@@@_
@@___@@_
@@___@@_
@@___@@_
________
```

Sleepy Ditto

Input File – sleepy.dat

General Statement – While traveling to the elite four, you accidently cross a trainer who challenges you to a battle. The trainer has only three Pokémon, and his first choice is a Jigglypuff. You know that Jigglypuff's song will make your Pokémon fall asleep. In order to conserve your party's strength for the elite four, you select your weakest Pokémon to battle the Jigglypuff….Ditto. Throughout the battle your Ditto is fading in and out of sleep and in his dazed state he keeps transforming into weird mutations of Pokémon. Your job is to identify when Ditto transforms into a mutation and tell him to wake up.

Input - There will only be three Pokémon that Ditto will have transform into. The data file will consist of six lines or three pairs of two lines. The first line in a pair will be name of the Pokémon Ditto is trying to transform into. The second will be the actual Pokémon ditto became.

Output – If Ditto correctly transforms print out: `Ditto got it right, good job Ditto!`
If ditto got it wrong print out: `That's not right! Ditto wake up!`

Assumptions:

- There will be exactly six lines (3 pairs of lines) in the data.
- The first of every other line is the correct Pokémon, and the second is what Ditto transforms into.

Sample Input:

```
Jigglypuff
Jugglypoof
Metapod
Metapod
Fearow
Freatrow
```

Sample Input:

```
That's not right! Ditto wake up!
Ditto got it right, good job Ditto!
That's not right! Ditto wake up!
```

Pokemon Catching Contest

Input File: contest.dat

Problem Description: Every month, there is Pokémon Catching Contest where trainers from all over the world come and hone their Pokémon catching abilities. Ash, feeling a little uneasy about his Pokémon catching prowess decides that this year he should participate in the Pokémon Catching Contest as well. The rules are simple. There is a certain number of Pokémon with each having different point values. With a limited number of Pokéballs, try to get the highest score. For each contest scenario, find the maximum score that Ash can get.

Input: The first line contains an integer N. The next N lines will be data sets. Each set contains 7 integers separated by a space. Each data set is formatted as follows
"A B C D E F G" where
A - the number of Type A Pokémon that can be caught
B - the number of Type B Pokémon that can be caught
C - the point value for catching a Type A Pokémon
D - the point value for catching a Type B Pokémon
E - number of poke balls needed to catch a Type A Pokémon
F - number of poke balls needed to catch a Type B Pokémon
G - the number of Pokéballs you have

Output: Output the most amount of points that can be obtained.

Assumptions: All Integers (A, B, C, D, E, F, G) will be between 0 and 50 inclusive.

Sample Input:

```
4
9 2 2 5 1 3 11
2 4 12 4 6 7 25
8 5 3 4 5 4 20
11 3 2 8 2 6 12
```

Sample Output:

```
21
28
20
16
```