Spring 2017 ACM Programming Contest

Overview

Welcome to the Spring 2017 ACM Programming Competition! This is a contest wherein you compete amongst your fellow students, and professors, in an effort to solve a set of programming problems as fast as possible.

Problems

There are eight problems of (roughly) increasing difficulty. The team who solves the most problems is declared the winner. In the event of a tie, the team with the best time wins. The best time also included penalties for wrong submissions. Each problem has a general description, details on the formats for input and output, and example input and output.

Input is done from standard input and output is done to standard output. *Do not open any files inside of your program.* Output of your program must match that of the correct output *exactly*. You can solve the problems in C, C++, Haskell, Java, PHP, Python, Racket, Ruby, or R. Up to date versions of these languages will be used to judge your entries. You should save your code using the standard extension for your language of choice (.c .cpp .hs .java .php .py .rkt .rb .R respectively).

At the end of your packet, there is also a brief overview of how to perform input and output in C++, Python and Java in case you need a refresher.

Rules

You are allowed to have any printed material such as books or printouts. You are also allowed to access standard library documentation for your language of choice. *No other use of the internet is permitted.* You are also not allowed to copy and paste code from any source be it a thumb drive, or another file that you have. Only one computer is allowed to a team.

Logging In

To log in to the system, direct your browser to `http://bit.ly/1NKMdFr`. You will then need to enter your team name and password which are provided for you. This will bring up the interface for you to submit solutions to the problems, ask questions (which will be broadcast to all participants), and see how your team is doing.

Submitting

To submit a solution, first make sure that you select the correct problem letter on the top of the site. Then click the "Choose File" button. Then choose the file that contains your source code (be sure it has the correct extension). After that it will be submitted for judging. It will receive an automatic judging response which is marked pending. Your solution will then be checked manually and a final response will be assigned.

DO NOT TURN OVER THIS SHEET UNTIL THE CONTEST STARTS

Problem A: Elixir of Life

Nicolas Flamel is the discoverer of the Philosopher's Stone, a "legendary substance with astonishing powers. The Stone will transform any metal into pure gold. It also produces the Elixir of Life, which will make the drinker immortal."

Nicolas and his wife Perenelle use the Stone to produce Elixir of Life which has kept them alive from the time of their birth in the 14th century up until the present day. When traveling, Flamel does not take the Stone with him because of the risk of losing it or having it stolen. Instead he makes enough Elixir to take with him for his trip. One liter lasts Flamel and his wife 12 days.

Flamel has asked you to write a program that will calculate how many liters of Elixir he should make in preparation of a trip lasting N days. For instance, if N is 15, then he would need 2 liters of Elixir.

Input

Input will consist of a single positive integer N giving the number of days the trip will last.

Output

Output will consist of a single integer giving the number of liters of Elixir Flamel will need for the trip. He only ever makes the Elixir in liter quantities, so your output must be an integer.

Sample Input

```
15
```

Sample Output

```
2
```

Problem B: Wizard Gold

Gringotts Wizarding Bank is a bank owned and operated by goblins. The bank is used by most in the wizarding world to safeguard their money and other valuables. One of the tellers at Gringotts, the goblin Griphook, has asked you to help him by writing a program that will figure out the best way to give change.

Wizarding money is broken down into three types of coins: Galleons, Sickles, and Knuts. The Galleon is worth the most. One Galleon is worth 17 Sickles, and one Sickle is worth 29 Knuts. So one Galleon is also worth $17 \times 29 = 493$ Knuts.

When giving change at Gringotts, Griphook prefers to use as many of the more valuable coins as possible. For instance, instead of giving a customer 200 Knuts, he would give them 6 Sickles and 26 Knuts. Griphook has asked you to write a program that is given some number of Galleons, Sickles, and Knuts, and outputs the same total value, but using as many valuable coins as possible.

Input

Input will consist of the three lines, each containing a single non-negative integer. The first is the number of Galleons, the second is the number of Sickles and the third is the number of Knuts. Taken together, they give the total value Griphook needs to give as change.

Output

Output will consist of three lines, each containing a singled non-negative integer. The first is the number of Galleons, the second is the number of Sickles and the third is the number of Knuts. Taken together, they should equal the same monetary value as the input, but use as many larger value coins as possible.

Sample Input

```
1
16
210
```

Sample Output

```
2
6
7
```

Problem C: Ancient Wizards

Hermione is very interested in reading about famous witches and wizards of the past. Recently she became interested in figuring out how long some of these magic users of antiquity lived.

Unfortunately, in many cases, the exact dates that these witches and wizards were born or died are not known for sure. For instance Merlin is know to have been born some time between the years 1050 and 1100, and to have died some time between 1225 and 1250. So Hermione concludes that Merlin lived at least 125 years and at most 200 years.

You will write a program that will automatically figure out the minimum and maximum age of a list of witches and wizards based on the earliest and latest dates that they are known to have been born or died.

Input

The first line of input is N giving the number of witches and wizard as input. Following that are N lines, one for each test case. Each of these lines starts with the name of the witch or wizard (containing no spaces), followed by 4 integers. The first two of these give the earliest and latest year the witch or wizard may have been born. The second two integers give the earliest and latest year they may have died.

Output

You should output N lines of output, one for each test case. Each of these should be in the format "NAME lived to between the ages of MIN and MAX.", where NAME is the name of the witch or wizard, and MIN and MAX are the minimum and maximum life span that witch or wizard could have lived.

Sample Input

```
3
Merlin 1050 1100 1225 1250
Gwydion -25 10 72 86
Circe -600 -575 -530 -530
```

Sample Output

```
Merlin lived to the age of 125 to 200.
Gwydion lived to the age of 62 to 111.
Circe lived to the age of 45 to 70.
```

Problem D: The Elder Wand

The Elder Wand is a legendary wand of immense power that is said to have been created by Death himself and given to the wizard Antioch Peverell. Because the wand is so powerful, it is coveted by wizards who will stop at nothing to get it. When the wand's owner is defeated, control of the wand passes to the victor. Peverell was murdered in his sleep which passed control of the wand to his killer.

If we know that the first owner of the Elder Wand is Peverell, then it is possible to trace the wand's ownership to the present day by going through lists duel results (which will include actual wizard duels as well as more underhanded tacts like Peverell's killer took).

Input

The first line of input will give the initial owner of the Elder Wand. The econdine of input will be a positive integer N giving the number of duel results. After that will be N lines each containing a wizard's name, followed by the word "defeats", followed by another wizard's name. Each name will be no more than 30 characters, and will contain no spaces.

Output

Output should be a single line stating "The Elder Wand belongs to Wizard" where "Wizard" is replaced by the new owner of the Elder Wand.

Sample Input

```
Peverell
11
Emeric defeats Peverell
Emeric defeats Sehaleus
Godelot defeats Emeric
Barnabas defeats Godelot
Loxias defeats Barnabas
Aflaviar defeats Azax
Gregorovitch defeats Loxias
Grindelwald defeats Gregorovitch
Dumbledore defeats Grindelwald
Malfoy defeats Dumbledore
Potter defeats Malfoy
```

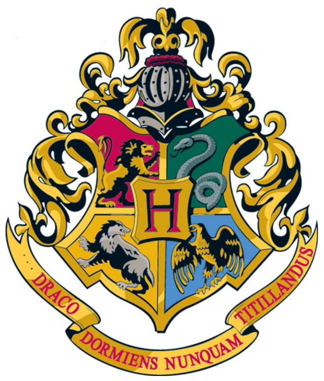
Sample Output

```
The Elder Wand belongs to Potter.
```

Problem E: House Sorting

Hogwarts School of Witchcraft and Wizardry contains 4 “houses”: Gryffindor, Hufflepuff, Ravenclaw and Slytherin. All students belong to one of these houses. Students in a house live and work together during their time at the school.

Students are “sorted” into one of the four houses when they first arrive to the school. Traditionally they are sorted by the Sorting Hat, which magically determines which house a student would fit in with best. Each of the four houses has a trait which the students have in common. For Gryffindor, it is bravery, for Hufflepuff it is dedication, for Ravenclaw it is intelligence and for Slytherin it is ambition. The student puts on the sorting hat and it judges which quality the student best exhibits and places them based on that.



Unfortunately, this year the sorting hat has gone missing! In its absence, the leaders of the school have determined that they will give each student a personality test to measure each of those four qualities. Each student will be assigned a grade for each from 0 to 100.

In order to sort the students, the following procedure will be used. First, we will calculate each student’s scores on those qualities as a percentage of their total points. For instance, if the student has a 60 for bravery, 45 for dedication, 55 for intelligence, and 40 for ambition, then we will divide each value by the student’s total points (200 in this case). This will give them scores of .3, .225, .275 and .2. This will normalize their scores for their overall aptitude.

Next, each house will “take turns” in a round-robin fashion, each selecting the remaining student with the highest normalized score in that house’s preferred quality. For instance Gryffindor will select the student with the highest score for bravery, while Hufflepuff will select the student with the highest score for dedication.

The houses will select students in alphabetical order, with Gryffindor starting and Slytherin finishing (the head of the Slytherin house was not happy about this). After each round, it will go back to Gryffindor’s turn again. This process will repeat until all of the students have been sorted.

In the event of a tie – where two students have an identical score in some category – the one whose name comes first alphabetically should be chosen.

The Headmaster of Hogwarts has hired you to write a program which will automatically sort all of the students using this procedure.

Input

The first line of input will be an integer *N* giving the number of students to be sorted. Following that will be *N* lines, one for each student. Each of these lines will begin with a string giving the last name of the student. After that will be four integers (in the range 0 through 100), giving the students scores for bravery, dedication, intelligence and ambition, in that order.

Output

The output should consist of the list of students assigned to each house. Each house name should be printed on a line by itself. Then each student in that house should be printed in alphabetical order. There should be a blank line between each house listing and the houses should be listed in the order Gryffindor, Hufflepuff, Ravenclaw, Slytherin.

Sample Input

```
8
Potter 95 90 80 70
Malfoy 50 40 80 90
Abbot 55 90 60 35
Goyle 25 15 20 60
Granger 90 85 90 80
Lovegood 70 70 85 25
Diggory 80 100 75 60
Chang 65 70 90 60
```

Sample Output

```
Gryffindor:
Granger
Potter

Hufflepuff:
Abbot
Diggory

Ravenclaw:
Chang
Lovegood

Slytherin:
Goyle
Malfoy
```

Problem F: Chamber of Secrets Part One

Harry is in the chamber of secrets deep under Hogwarts. He is in search of the Basilisk, a fearsome creature also known as the King of Serpents. Harry starts in the upper level of the chamber. Before he can deal with the Basilisk itself, he will need to find the stairs which lead down to the lower level of the chamber.

The chamber of secrets consists of several rooms with paths connecting them. Harry will begin some place in the chamber and have to navigate to the stairs. It is possible that there will not be a path. Your job is to write a program which will check if there is a path which will take Harry from his starting position to the stairs.

Input

The first line of input will be an integer, C, giving the number of columns of the chamber. The second line will be an integer, R, giving the number of rows of the chamber. Following that will be R lines each containing C characters, giving the layout of the chamber.

Each character will be one of:

- A **space** character, indicating a path or part of an open room.
- A # character, indicating solid wall which Harry can not walk through.
- An **H** character, indicating Harry's starting position in the chamber. There will be exactly one of these.
- An **S** character, indicating the location of the stairs. There will be exactly one of these.

Output

Output should be one line, either "Harry can reach the stairs!" if there is a path between Harry and the stairs, or "Harry can not reach the stairs!", if there is not.

Sample Input 1

```
40
20
########################################
#         ##############################
# H                                  ###
#         ##################### #### ###
#         #####                 #### ###
#### ########################## #### ###
#### ########################## #### ###
#### ###                        #### ###
#### ### ###################### #### ###
#### ### ####            ###### #### ###
#### ### #### ########## ###### #### ###
#### ##   ### ##      ## ###### #### ###
#### ## S ### ##      ## ##       ## ###
#### ##   ### ##         ##       ## ###
#### ######## #############       ## ###
####                    ###       ## ###
###########################       ## ###
#################################### ###
######                               ###
########################################
```

Sample Output 1

```
Harry can reach the stairs!
```

Sample Input 2

```
40
20
########################################
#         ##############################
# H                                  ###
#         ##################### #### ###
#         #####                 #### ###
#### ########################## #### ###
# ## ########################## #### ###
# ## ###                      # #### ###
# ## ### ###################### #### ###
# ## ### ####                   #### ###
# ## ### #### ########## ###### #### ###
# ## ##   ### ##      ## ###### #### ###
# ## ## S ### ##      ## ##       ## ###
# ## ##   ### ##         ##       ## ###
# ## ### #### #############       ## ###
# ###### ####           ###       ## ###
#        ##################       ## ###
#        ###############################
#                                    ###
########################################
```

Sample Output 2

```
Harry can not reach the stairs!
```

Problem G: Chamber of Secrets Part Two

Harry is now in the lower level of the chamber of secrets, again trying to find his way. This time he is looking for the legendary sword of Gryffindor. He will need the sword in order to deal with the Basilisk. As before, he has to navigate the paths of the chamber looking for his goal.

However, somewhere in this level of the maze is the Basilisk itself! The Basilisk is a deadly foe, not only can it attack Harry with its venomous fangs, but looking into its eyes directly causes instant death. Looking into its eyes indirectly, such as reflected off a mirror causes petrification. The heir of Slytherin has placed angled mirrors throughout the chamber to make traversing the paths even more difficult.

Harry must find his way through the chamber to the sword, without ever looking into the Basilisk's eyes - either directly or reflected through a mirror. Your goal is to write a program which will test if there is a path from Harry's starting location to the sword without ever looking at the Basilisk or the Basilisk's reflection.

Harry's direction is now important. For instance, he can walk *past* the gaze of the Basilisk, as depicted below:

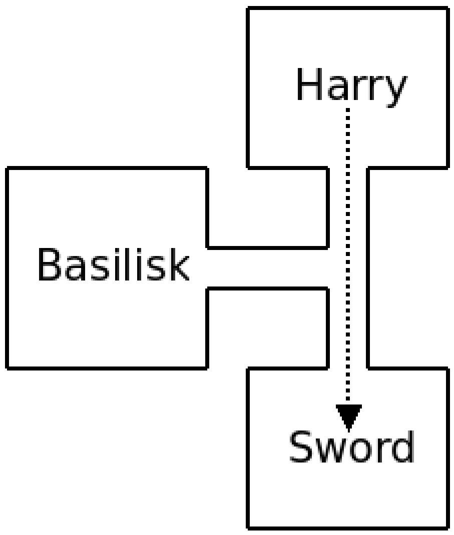


Figure 1: Walking past the Basilisk is OK

Harry can reach the sword in this chamber because he doesn't face the gaze of the Basilisk, he passes by it with it on his left – Harry always faces the direction he is moving.

Harry also can walk with the gaze of the Basilisk at his back, as depicted in the following diagram:

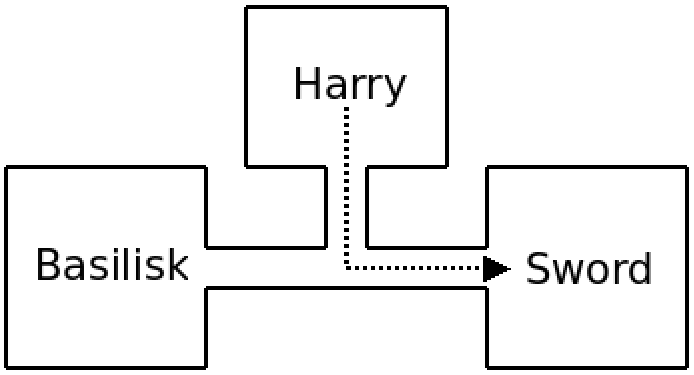


Figure 2: Walking with the Basilisk at our backs is OK

Here, Harry can turn to his left when he reaches the bottom of the center hallway, and continue in that direction until he reaches the sword. The gaze of the Basilisk is only deadly when you face it as shown below:

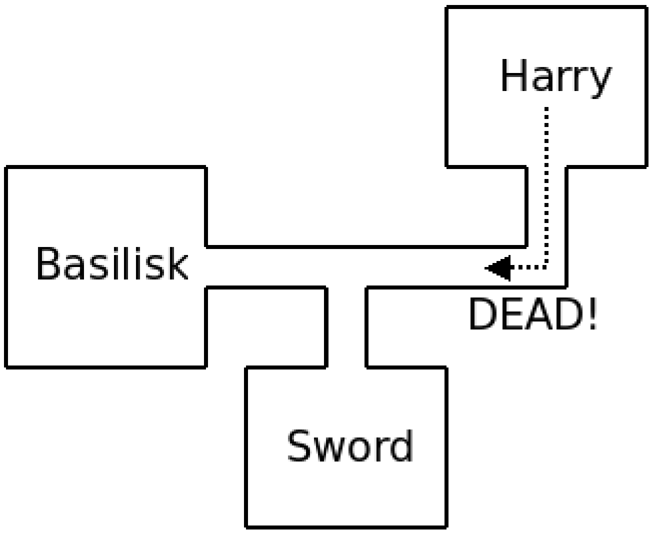


Figure 3: Staring directly at the Basilisk results in death!

Here, Harry can not reach the sword because he will have to face the Basilisk directly to do so, which will kill him. As mentioned previously, there are also angled mirrors which reflect the gaze of the Basilisk. An example is shown below:

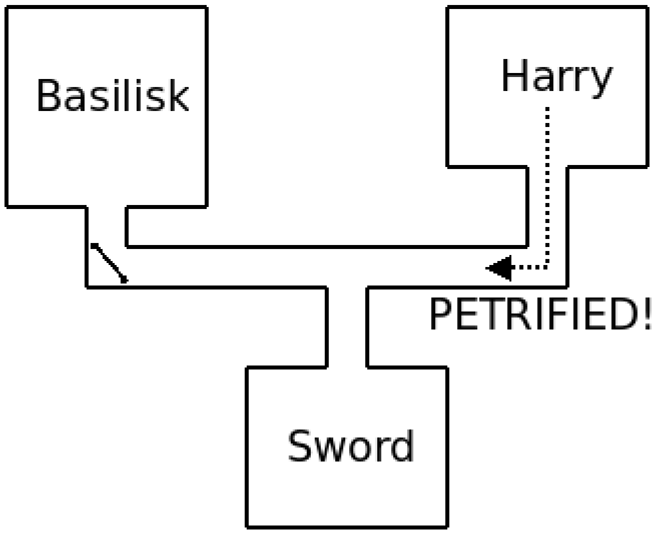


Figure 4: Staring at the Basilisk through a mirror results in petrification!

Here, Harry can also not reach the sword, because when he turns to his right, he will see the gaze of the Basilisk reflected off the angled mirror at the end of the long hallway.

Note that it is possible that the gaze of the Basilisk can reflect off of several mirrors in series before reach Harry.

Your goal is to see if there is at least one path through the chamber by which Harry can reach the sword without dieing or being petrified.

Input

The first line of input will be an integer, C, giving the number of columns of the chamber. The second line will be an integer, R, giving the number of rows of the chamber. Following that will be R lines each containing C characters, giving the layout of the chamber.

Each character will be one of:

- A **space** character, indicating a path or part of an open room.
- A # character, indicating solid wall which Harry can not walk through.
- An **H** character, indicating Harry's starting position in the chamber. There will be exactly one of these.
- A **B** character, indicating the location of the Basilisk. The Basilisk does not move. There will be exactly one of these.
- A \ or / character, indicating an angled mirror. The gaze of the Basilisk reflects off the angled mirrors in the directions you would expect. For instance, if the Basilisk is to the left of a \ mirror, the gaze will be reflected down, whereas if he is to the left off a / mirror, it will be reflected up. Harry cannot walk over a mirror.
- An **S** character, indicating the location of the sword. There will be exactly one of these.

Output

Output should be one line, either "Harry can reach the sword!" if there is a path by which Harry can reach the sword, or "Harry can not reach the sword!", if there is not.

Sample Input 1

```
40
20
########################################
#H        ##############################
#                                   ####
#         ########################  ####
#         #########################   ##
###### ############################## ##
###/                               \# ##
###  # ############# #############  # ##
### ## ############# ############## # ##
### ## ############# ##############   ##
### ## ############# ############## ####
### ##               ############## ####
### ################ #######        ####
### ################ ####### ###### ####
### ################ ####### ###### ####
### ################        S  #       #
###  ###########################       #
###\                               B   #
################################       #
########################################
```

Sample Output 1

```
Harry can reach the sword!
```

Sample Input 2

```
40
20
########################################
#H        ##############################
#                                   ####
#         ########################  ####
#         #########################   ##
######## ############################ ##
###/                               \# ##
###  # ############# #############  # ##
### ## ############# ############## # ##
### ## ############# ##############   ##
### ## ############# ############## ####
### ##               ############## ####
### ################ #######        ####
### ################ ####### ###### ####
### ################        S  #       #
###  ###########################       #
###\                               B   #
################################       #
########################################
```

Sample Output 2

```
Harry can not reach the sword!
```

Problem H: Animagus Transformations

Sirius Black is an *Animagus*: a wizard who is able to transform into an animal at will. In Sirius Black's case, he is able to transform into a large dog.

While in dog form, Sirius is able to run faster than he normally can. However, he is able to open doors much faster as a human (due to having human hands). He also takes some amount of time to transform back and forth.

Sirius is on the run from the department of magical law enforcement, who still believe him to have murdered Peter Pettigrew. He needs to escape along a hallway which contains some number of doors. He wants to know what the fastest way to do it would be: should he do it as a human who can open doors faster, as a dog who can run faster, or would it be best to transform back and forth as he goes?

As a dog, Sirius can run 10 meters per second, while he can only run 4 meters per second normally. It takes Sirius 8 seconds to open a door in dog form, and only 1 second normally. It takes him 5 seconds to transform to and from dog form.

You should write a program that, given the distance Sirius needs to travel, and the locations of all of the doors in that distance, reports the shortest time he could cover the distance, along with how many transformations he would make.

Sirius always begins in normal, human form.

Input

The first line of input is the distance, in meters that Sirius needs to travel. The second line will contain the number of doors. There will be between 1 and 15 doors. The third line contains the locations of the doors along that path, separated by spaces.

Output

Your output should be of the form:

"Sirius could travel this distance in X seconds, transforming Y times."

Where X is the minimum number of seconds that Sirius could travel the distance and Y is the number of times he would need to transform to do this. If Y is equal to 1, you should output

“1 time” instead of “1 times”.

Small rounding errors in the time will be accepted for this problem!

Sample Input

```
100.0
4
10.0 80.0 85.0 90.0
```

Sample Output

```
Sirius could travel this distance in 28.500 seconds, transforming 2 times.
```

Doing Input and Output

This guide contains the basic way of doing input and output in C++, Java, and Python, in case you need a quick refresher.

In programming contests, you should always do input from the terminal screen (e.g. using the keyboard), and do output to the terminal screen. You should not open any files for input or output.

You should also not output any prompts. If a problem instructs you to read a line containing a number, just read it in without any kind of prompt saying "Enter a number" or similar.

C++

Input

1. To skip over whitespace (spaces or new lines), and read in a single value (such as an integer or string), use
   `cin > > value;`. For example:

   ```
   // read one integer
   int number;
   cin >> number;

   // read one character string
   char str1[100];
   cin >> str1;

   // read one string object
   string str2;
   cin >> str2;
   ```

2. To read in one entire line of input, which may contain spaces, use `cin.getline`. For example:

   ```
   // read in a character string of up to 100 characters
   char str1[100];
   cin.getline(str1, 100)

   // read in a string object
   string str2;
   getline(cin, str2);
   ```

Output

Output is done with `cout < <` in C++ which can take any built-in data type. For example:

```
// print a message, then an integer, then a new line
int x;
cout << "X is equal to " << x << endl;
```

Python

Input

1. The `input` function in Python read in one line of input and returns it as a string. For example:

   ```
   # read in a string
   line = input()
   ```

2. In order to convert from a string to a number, you can use the `int` function for integers, or the `float` function for real numbers:

   ```
   # read in an integer by passing the input string to int()
   number = int(input())
   ```

3. In order to break a line of input into multiple strings, separated by spaces, you can use the `.split()` function which returns a list of strings:

   ```
   # read in a whole line
   line = input()

   # split it into separate strings based on spaces
   words = line.split()
   ```

Output

1. Output in Python is done with the `print` function which outputs all of its arguments, separated by spaces, and puts a newline at the end:

   ```
   # print a message, a number and a new line
   print("X is equal to", x)
   ```

2. In order to prevent `print` from putting spaces between each item, pass `sep=' '` as an argument:

   ```
   # now there is no space between message and value
   print("X is equal to", x, sep= )
   ```

3. In order to prevent `print` from putting a new line at the end, pass `end=' '` as an argument:

   ```
   # now there is no new line added
   print("X is equal to", x, end= )
   ```

Java

Input

Input in Java can be done with the Scanner class which must be imported first:

```
import java.util.Scanner;
```

Then a Scanner object must be created:

```
java.util.Scanner in = new java.util.Scanner(System.in);
```

1. To read in one line of input into a string, use the scanner's nextLine method:

   ```
   String line = in.nextLine();
   ```

2. To read in a single word into a String, stopping at a space, use the next method:

   ```
   String word = in.next();
   ```

3. To read a numerical value use the nextInt method for integers, or the nextDouble method for real numbers:

   ```
   int number1 = in.nextInt();
   double number2 = in.nextDouble();
   ```

Output

1. To output a string constant or variable, use the System.out.println function which takes one argument, prints it to the screen, then prints a new line:

   ```
   // print a message
   System.out.println("This will be printed")

   // print a value
   int x;
   System.out.println(x);
   ```

2. To output something without a new line at the end, use the System.out.print function which outputs its argument with no newline:

   ```
   // print a message with no new line
   System.out.print("The value of X is ")

   // print a value with no newline
   int x;
   System.out.print(x);
   ```