Casino I

Input file:	`standard input`
Output file:	`standard output`
Time limit:	12 seconds
Memory limit:	256 megabytes

This is an interactive problem.

Vertex Cover team got tired of problem solving and decided to go to the casino to have some fun and play Blackjack. In this casino, the goal of Blackjack is to collect cards that total to more points than the dealer, **without exceeding** 1.

Vertex Cover members are not good in betting, so they asked for your help.

You will play several rounds, In each round, each one of you and the dealer have initially a score of 0, the dealer will ask you if you want a card with a randomly and **uniformly** generated number between 0 and 1 written on it. If you say *hit*, the dealer will give you that card. If your total hand (the sum of numbers written on the cards) becomes **more than** 1, you get *bust* and lose. Otherwise, the dealer will ask you again. If you say *stand*, your turn is over with a total T, and dealer's turn starts. The dealer will get a card and do the following :

1. If his total points is less than or equal to T he will get another card and repeat again.
2. If his total points is strictly more than 1, you win the round.
3. If his total points is strictly more than T but no more than 1, you lose the round.

These are the rules, You have to play $n = 10^5$ rounds, if you win at least 42% of the rounds, you will win and get the mega millions jackpot. Good luck !

Input

A single line containing a single integer $n = 10^5$: The number of games.

Interaction Protocol

After reading the number of games n. You will play 10^5 rounds, each round starts when the interactor outputs G.

In each turn of the **current** round, you have two possible movements:

- 1 : The interactor will generate a uniformly random variable between 0 and 1, and adds it to the sum.
  - If the current sum exceeds 1, the interactor will output L
  - Otherwise, it will output "S r" where r is the current sum.
- 0: Your turn is over, and the dealer will play **optimally**. Then the interactor will give you the winner of the round as follow:
  - It will output L if you lose.
  - It will output W if you win.

You have to use a flush operation right after the movement. For example, in C++ you should use the function *fflush(stdout)* or *cout.flush()*, in Java — *System.out.flush()*, and in Python — *sys.stdout.flush()*.

Example

standard input	standard output

3
1 1 1 0
1 1 1 1 0
1 1 1
	G
	S 0.265346
	S 0.789890
	S 0.932484
	W
	G
	S 0.070631
	S 0.182233
	S 0.726913
	S 0.827251
	L
	G
	S 0.785957
	S 0.896195
	L

Note

- There's no limit to how many cards you can ask for, but once your hand total's higher than 1, you bust and the dealer gets your bet.
- The example above is **not a complete** example, as $n = 3$, and not $n = 10^5$, but it is included **to understand** how the game should work.
  - For the first game, you played 3 turns,less or equal than reaching a sum $S = 0.932484 \leq 1$. And with that, you skipped.
  - For the second game, you played 4 turns, reaching a sum $S = 0.827251 \leq 1$. And with that, you skipped
  - For the third game, you played 3 turns, but you reached a sum $S > 1$. This can be viewed as you lost before skipping.
- As the dealer's sum is not seen, and will not be either in other test cases, you **cannot see** how you won/lost. We will detail the scenarios:
  - For the first game, the dealer got $S_{\text{dealer}} = 0.549321$ after the first turn, and then $S_{\text{dealer}} = 0.86321$. Finally he took a card again, and got $S_{\text{dealer}} = 1.332513 > 1$, and so he lost, and you **won**.
  - For the second game, the dealer got $S_{\text{dealer}} = 0.836332 > S = 0.827251$ after the first turn, and so he won, and you **lost**.
  - In the third game, After getting $S = 0.896195$, but you chose to take a card again, and you got $S = 1.62371 > 1$. And with that, you **lost**.