# (A) Raju's Conlang

## Problem Description

As we all know Python was created by Guido van Rossum, a Dutch programmer. Raju a coding enthusiast, is trying to become a new Guido van Rossum, by developing a new coding language!

Initially, he started with developing syntaxes, especially the *if*, *if-else* and *nested if*, which are control flow structures and only using comparison operators(>, <, ==, !=) along with the print function. He felt that it should deviate from the syntax what already exists in other languages. So he made it like below -

*If* Syntax in Raju's coding language

```
is A<B
Yes
print A
si
```

*If* - *Else* Syntax in Raju's coding language to find the largest among two numbers.

```
is A<B
Yes
print B
No
print A
si
```

Nested If syntax in Raju's coding language to find the largest among three numbers.

```
is A>B
Yes
is A>C
Yes
print A
No
print C
si
No
is B>C
```

Yes

print B

No

print C

si

Given a script, just having either *If, If-else* or *nested If*, print the output.

## Constraints

3 <= number of lines in the script <= 50

1 <= values that variables hold <= 50

## Input

The script with any of *if, if-else , nested if* will be present in the first few lines.

Second last line has the variables used in the script, separated by space. The variable names will be A to Z.

Last line consists of space separated integers denoting the values of variables mentioned in the above line.

## Output

Print the output of the script.

## Time Limit (secs)

1

## Examples

Example 1

Input

is A<B

Yes

print A

si

is A>B

Yes

print B

No

print A

si

A B

1 2

Output

1

1

Explanation

The above script depicts -

```
if (A < B){
print(A)
}
if (A > B){
print(B)
}
else{
print(A)
}
```

Here A=1 and B=2, since B > A, according to the condition, we print A as output in the first line which is 1, followed by A>B ,by condition executing else block and printing A as output in the second line which is 1.

Example 2

Input

is A<B

Yes

is A!=C

Yes

print A

si

No

print B

si

A B C

1 2 3

Output

1

Explanation

The above script depicts -

```
if (A < B){

if (A != C){

print(A)

}

}

else{

print(B)

}
```

The values of A, B, C are 1, 2, 3 respectively. According to the given conditions, we print 1 as output.

# (B) Getaway Gala

## Problem Description

At the annual office holiday party, excitement buzzed as employees gathered for a lucky draw. Among the prizes was a weekend getaway voucher.

With a small number of attendees forming a row, the organizer chose to forego the conventional lucky dip approach. Instead, they introduced a more curious and engaging method that will eliminate every employee except one, who is considered as winner.

The organizer aimed to eliminate employees in two rounds. In the first round, he will form a row of employees who participated. Then he will select the first letters of all the employees from left to right and form a string S. Then he will start eliminating the employees whose name is forming the palindrome (every time he will delete the first name from left to right which is forming a palindrome). He will continue this process till the string S does not have any sub strings which are palindromes.

For example, consider the names {Hari Giri Siri Gopi Hima} in the row. Here the string S formed by picking the first letter of the names is HGSGH. Note that substring GSG is a palindrome .When processing from left to right, the alphabet G from the name Gopi forms a palindrome. We call the name *Gopi* as *Mirror Word*. Hence we eliminate Gopi in the first iteration.

Now only names {Hari Giri Siri Hima} remain. String S is HGSH which does not have any palindrome. Hence the final row will be {Hari Giri Siri Hima}. If string S had more palindromes we would apply the same procedure as mentioned in paragraph above. Since string S is now palindrome free, the organizers will now apply a different criteria.

This criteria will be to remove every N$^{th}$ person from the remaining names everytime processing the names from left to right. The last name remaining is the winner of the lucky draw.

Given the list of names of employees, and the value of N, find out the who the winner is.

## Constraints

Names comprise of upper and lower case characters. Processing is case insensitive.

1 <= number of employees <= 1000

1 <= N <= 1000

1 <= length of name <= 10

## Input

First line consists of an array indicating the names of employees present in the row. Names are space separated.

Second line consists of a single integer N denoting the interval of elimination.

## Output

Print the name of the employee who is going to be the winner. Print the name as it is given in the array.

## Time Limit (secs)

1

## Examples

Example 1

Input

Janu gita sana gopi jaslin Tony Ritu Naina sonu Neha

2

Output

Janu

Explanation

As we can see, gopi is forming a mirror word according to the given rules, thus we eliminate the employee named gopi. Now gopi's position will be empty. Again, the name Neha is forming mirror word, hence we remove the name Neha and the resulting row will be {Janu gita sana jaslin Tony Ritu Naina sonu}. Now there can be no palindromes after picking the first alphabets of the remaining names. Hence the second criteria is now applied where N is 2. Now, processing from left to right, if we start removing every 2nd name (Nth name) until we are left with one person, then it turns out that Janu will be the winner, hence print "Janu".

Example 2

Input

Vishal ram Vina veda Lekha Rahul Keerthi

1

Output

Keerthi

Explanation

As we can see, Vina is forming a mirror word, removing it will result in {Vishal ram veda Lekha Rahul Keerthi}. Next, veda forms the mirror word. On removing we will get {Vishal ram Lekha Rahul Keerthi}. Next, Rahul forms the mirror word which upon removal will yeild {Vishal ram Lekha Keerthi}. Now there can be no palindromes after picking the first alphabets of the remaining names. Hence the second criteria is now applied where N is 1.From Vishal, if we start removing every 1st name until we are left with one person, it turns out that Keerthi will be the winner, hence print "Keerthi".

# (C) SubString Quest

## Problem Description

Raju and Rani like playing with strings, exploring the interesting patterns of smaller parts within them, called substrings.

Presently, they are engrossed in a string-centric game and it is Raju's turn. His task involves two strings provided by Rani, namely S1 and S2. Raju is tasked with dividing the first string S1 into the fewest possible substrings, ensuring that all resulting substrings, whether in their original or reversed order, are present in S2 while also aiming for maximizing the length of current substring.

For example, if S1 = "vanaja" and S2 = "rvanpnaja", then the output will be van|aja rather than va|naja.

**Note** : The letter in the $i^{th}$ position in S1 can be used in exactly one substring of S1. The letters from S2 can be reused any number of times but the letters from S1 can be used only once. The letters are case sensitive in both the strings.

## Constraints

1 <= S1, S2 <= 10000

The two strings S1, S2 consist of lower and upper case characters only.

## Input

First line consists of a string S1.

Second line consists of string S2.

## Output

Print the string S1 as a concatenation of sub strings separated by '|'.

## Time Limit (secs)

1

## Examples

Example 1

Input

VaisHnavI

VIvekanandaJasHnavi

Output

V|a|i|sHnav|I

Explanation

Every sub string that is present in V|a|i|sHnav|I is also present in S2 in original or reversed order. No other way of doing sub strings on S1 will give less number of sub strings than this. Hence, we print V|a|i|sHnav|I.

Example 2

Input

sudHanShu

qniwprvAvnaHSsH

Output

Impossible

Explanation

As we can see, its impossible to sub string the string S1 based given conditions, thus we print "Impossible".

# (D) Clash Of Clans

## Problem Description

In Clash of Clans, you as a village chief is gearing up for a raid on an enemy village. The success of this mission depends on assembling the most potent raiding army by selecting troops from various categories. Each troop type has distinct strengths and weaknesses, and your goal is to create a formidable army that maximizes overall damage.

To build an army you first need to train them in the "Barracks" which has some capacity (B). Each troop has some size (S) it occupies in the barrack that trains and increases some amount of damage per second (D) for each troop.

There are various troops (for e.g. Barbarian, Archer, Giant, Goblin and so on) which belong to some category C (for e.g. Elixir Troop, Temporary Troop, Super Troops and so on).

To train them you decided to have a versatile army where you select at most one or no troops from each category of the troops such that it has maximum damage per second and the troops fit within the barrack size for training.

## Constraints

Length of S = D = C

1 <= length of S, D <= 100

1 <= Number of categories <= 20

1 <= B <= Sum of S

Size of the troop <= Size of the Barrack i.e. Si <= B

## Input

The first line contains the list of integers denoting damage per second capability $D_i$ of the troop.

The second line contains the list of integers denoting the size $S_i$ of the troop.

The third line contains a list of integers denoting the category $C_i$ of the troop.

Last line contains an integer denoting the size of the barrack.

## Output

Print the maximum damage per second that can be achieved.

## Time Limit (secs)

1

## Examples

Example 1

Input

8 9 4 9 1 8 1 5 6 8

2 5 7 2 3 4 5 9 3 8

4 2 2 3 4 3 2 1 2 1

10

Output

26

Explanation

The goal is to maximize the damage per second where you select at most one or none from each category. So here we choose the 1st troop which belongs to category 4, 2nd troop which belongs to category 2 and 4th troop which belongs to category 3 whose total size is 9 which is within the barrack size. We could not accommodate any troop from category 1 because the damage per second capability reduces or the barrack capacity falls short. Hence, the total damage per second is 26.

Example 2

Input

6 8 2 1 5 1 2 2 3 10

5 9 2 2 1 2 2 3 3 4

1 3 4 4 3 1 2 2 3 4

10

Output

21

Explanation

Here we choose the 1st troop which belongs to category 1, 5th troop which belongs to category 3 and 10th troop which belongs to category 4 whose total size is 9 which is within the barrack size. We could not accommodate any troop from category 2 because the damage per second capability reduces or the barrack capacity falls short. Hence, the total damage per second is 21.

# (E) Christmas Tower

## Problem Description

Daiwik received an exciting Christmas present from his Secret Santa. The present consists of multiple repeated blocks, each varying in size. There are five unique blocks, all with identical width but varying heights. Daiwik's task is to construct a tower using these blocks, ensuring he uses the maximum number of blocks and the overall height does not surpass a specified height H.

Given that Daiwik has an infinite supply of each block type, print the block heights prioritizing the frequencies in descending order, at which they were utilized in constructing the tower. If one or more type of blocks are used with same frequency, then print them in the ascending order of their heights.

## Constraints

1 <= L <= 10^4

1 <= heights of five blocks <= 10^3

## Input

First line consists of L denoting the height of the tower that Daiwik is going to build.

Next line consists of five space separated integers denoting the heights of the blocks that Daiwik have.

## Output

Print the block heights in descending order based on their frequency of usage in constructing the tower, separated by space. If one or more type of blocks are used with same frequency, then print them in the ascending order of their heights. Print "Impossible" if its not possible to construct a tower of height H.

## Time Limit (secs)

1

## Examples

Example 1

Input

13

2 7 3 4 9

Output

2 3

Explanation

To construct a tower with a height of 13, utilizing blocks of sizes {2, 7, 3, 4, 9}, Daiwik can opt for five blocks with a height 2 and a single block with a height of 3. This configuration would yield a total height of 2 * 5 + 3 * 1 = 13.

Example 2

Input

19

2 4 6 8 10

Output

Impossible

Explanation

Constructing a tower of height 19 using the provided combination of blocks is not feasible. Hence, print "Impossible".

# (F) Room Optimization

## Problem Description

Vishal, a builder focused on constructing rooms along the port side, acquires land with the goal of optimizing the number of rooms he can construct with that space. The rooms are limited to four specific types, categorized as k-square unit rooms, where k can take on one of the values {A, B, C, D}.

At first, Vishal had the opportunity to purchase X units of land. Subsequently, he had the chance to acquire an additional Y units of land, which is contiguous to the initial plot.

He follows two rules while constructing which are given below.

- He aims for constructing more number of rooms within that area, irrespective of the size of the rooms.
- If there are multiple combinations of rooms possible within given area and maximum number of rooms is same for all combinations, then he will opt for the combination with larger room area. For example if the area is 21 units, and the values of {A, B, C, D} are {5, 9, 7, 11} then we have (7, 7, 7), (9, 7, 5), (11, 5, 5). Here the combination (11, 5, 5) is comprises of three rooms with largest room size be 11.since Vishal prefers larger rooms given the total number of rooms is same, this combination will be constructed.

Given the values of {A, B, C, D}, the initial land Vishal bought and the additional units of land he acquired, calculate the difference in the number of rooms of each type resulting from the additional Y units of land.

## Constraints

1 <= A, B, C, D <= 1000

A, B, C, D will be Unique.

1 <= (X+Y) <= 1000

## Input

First line consists of four space separated integers denoting the values of A, B, C and D.

Second line consists of two space separated integers denoting the values of X and Y.

## Output

Four space separated integers denoting the difference in the number of rooms of each type in the same order given in the input. Note that the difference will be calculated as (number of rooms of Type-N before extension) - (number of rooms of Type-N after extension).

## Time Limit (secs)

1

## Examples

Example 1

Input

9 11 7 5

21 6

Output

0 1 -1 -2

Explanation

Given initial area is 21 units. Vishal can plan to build 1 room of 11 square units and 2 rooms of 5 square units which will occupy 1 * 11 + 2 * 5 = 21 square units. But the extended area is 6 units, which means the total area in which he can build rooms is 21 + 6 = 27 square units. For maximizing the number of rooms within 27 square units, he can build 1 room of 7 square units and 4 rooms of 5 square units which will occupy 1 * 7 + 4 * 5 = 27 square units.

Hence the final composition of rooms before and after extension is -

| Room Type | Number of rooms selected (Before Extension) |
|---|---|
| 9 square units | 0 |
| 11 square units | 1 |
| 7 square units | 0 |
| 5 square units | 2 |

Hence the difference in the number of rooms of each type is 0 1 -1 -2.

Example 2

Input

2 5 13 6

14 9

Output

-2 -1 0 0

Explanation

Given initial area is 14 units. Vishal can plan to build 7 rooms of 2 square units which will occupy 7 * 2 = 14 square units. But the extended area is 9 square units, which means the total area in which he can build rooms is 14 + 9 = 23 square units. For maximizing the number of rooms within 23 square units, he can build 9 rooms of 2 square units and 1 room of 5 square units which will occupy 9 * 2 + 1 * 5 = 23 square units.

Hence the final composition of rooms before and after extension is -

| Room Type | Number of rooms selected (Before Extension) |
|---|---|
| 2 square units | 7 |
| 5 square units | 0 |
| 13 square units | 0 |
| 6 square units | 0 |

Hence the difference in the number of rooms of each type is -2 -1 0 0

# (G) Bob The Beaver

## Problem Description

"Wow It is good place to build a Dam .Have enough logs also.lets start the work... ugh i need a blueprint" - Bob The Beaver

Bob decided to build a dam across the river. Bob has made up his mind to use odd number of logs for reasons best known to him. Logs can be of different lengths. The river flows straight and its width is constant. The length of logs and width of the river will be known before construction of the dam.

The Dam divided into three parts left , middle and right. Since the logs count is an odd number the middle log will always divides the dam into two parts.

- Log(s) from start to middle is left part of the dam,
- Middle log is the middle part of the dam and
- Log(s) from middle to last is the right part of the dam.

Although dam construction is a 3D activity we will consider only 2D aspects of the dam.

The dam construction guidelines are as follows

- Draw a imaginary line between the shores. This line should be perpendicular to the flow of river. This line is reffered as dam construction line.
- Log placing starts form the point where the dam construction line touches the left shore.
- The logs will be placed one after another in the order given in the input.
- The internal angle between logs placed in the left part of the dam and the dam construction line should be less 90 degrees.
- The log in middle part / the middle log has to placed parallel to dam construction line.
- The internal angle between logs placed in the right part of the dam and the dam construction line should be less that 90 degrees.
- After placing all the logs you have to reach the right point where dam construction line and right shore meets.

Given N logs and their lengths find the angles which the logs can be placed at, obeying the guidelines as stated above. Bob will place the log one by one according to the given angle. Finally he should reach the right shore point.The distance between the place bob reached on right shore and actual right shore point of the dam is considered as *Error*.

Bob has Error tolerance of +/- 0.10 for the distance between the actual and the obtained points on the right shore.

## Constraints

$2 < N < 30$

## Input

First line contains a single integer N representing the number of logs.

Second line contains N space separated integers representing the length of the logs.

Third line contains a single integer W representing the width of river.

## Output

Print the internal angles in degrees separated by space, corresponding to the logs. Restrict the angles upto two decimal points.

## Time Limit (secs)

1

## Examples

Example 1

Input

3

1 2 1

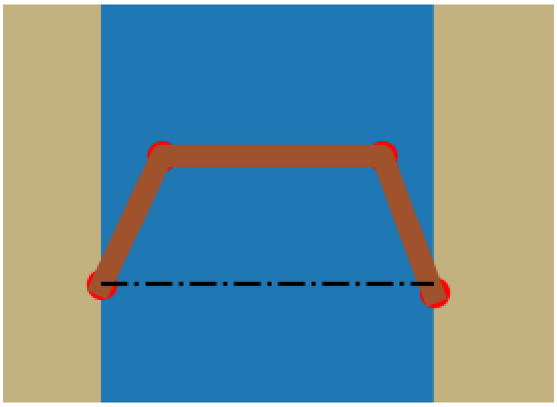
3

Output

57.16 0.00 62.89

Explanation

Lets take the left end of the dam construction line as (0, 0) and the right end as (3, 0) since the width of river is 3. The first log is kept at internal angle of 57.16 degree at (0, 0). The other end of this log will be approximately at (0.54, 0.84). From this point second log will be placed. Since this is the mid log it has to be kept parallel to the dam construction line i.e., 0 deg. This log will end at approx (2.54, 0.84). Then the third log will placed at internal angle of 62.89 . Then third log will end at (3.00, -0.05) . Hence the error is 0.05 which is within the tolerance limit.



Example 2

Input

5

1 2 2 2 1

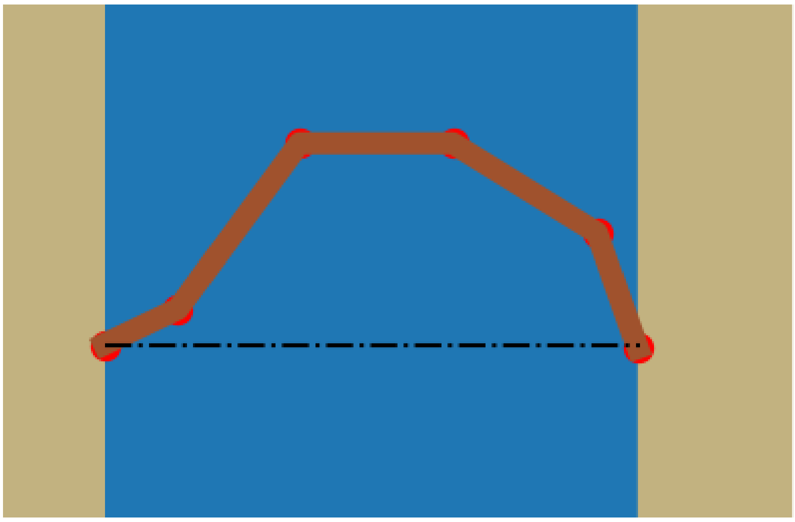
7

Output

15.26 37.78 0.00 19.41 58.23

Explanation

The below diagram in description will decpicts this inputs.



Assume the starting point as (0,0). Draw a imaginary line between the shore. The width of the river is 7. So the end point should be (7,0) . If logs are placed in the given order and angles as mentioned in the output, Bob will reach the point (6.96,-0.03) . For this point the error is 0.049 . which is below the tolerance limit of 0.10.

# (H) Dream11

## Problem Description

Cricket is a team sport comprised of 11 player each team has four types of player viz. Batsman, Bowler, All-Rounder, and Wicket Keeper. Dream11 is a fantasy cricket platform where users create their own cricket teams. You have decided to participate in Dream11 during the IPL season. The platform rewards prizes based on the maximum total value of your team. Your task is to create a cricket team based on certain constraints to win the reward.

Write a program to do so while adhering to the following constraints:

- Your team must consist of 11 players.
- You have a budget of B to spend on selecting players.
- Each player has a price tag and a player value.
- There are four types of player: Batsman, Bowler, All-Rounder, and Wicket Keeper.
- Your team should have at least one Wicket Keeper, one All-Rounder, two Batsmen, two Bowlers.

Now given list of price and player values to determine the type of players:

- The first 20% of players are considered Wicket Keepers. Note: Take the ceil of the number obtained.
- Batsmen are selected from odd positions (excluding the first 20%).
- Bowlers are selected from the even positions that are not divisible by 4 (excluding the first 20%).
- All-Rounders are selected from positions divisible by 4 (excluding the first 20%).
- Player index starts from zero. Please factor this in calculation of player types viz. Wicket Keeper, All-Rounder, Batsmen and Bowler. *Refer example section for better clarity.*

## Constraints

11 <= N <= 200

1 <= B <= 1000

1 <= P < = 20

1 <= V < = 20

## Input

The first line contains N the total number of players.

The second line consists of vector P which denotes the list of player prices, where the $i^{th}$ element represents the price of $i^{th}$ player.

Third line consists of vector V which denotes the list of player value, where the $i^{th}$ element represents the values of $i^{th}$ player.

The last line contains B the total budget.

## Output

Print the maximum total value M which is the summation of the selected player values, that can be obtained while satisfying all the constraints and is within the budget else print "Insufficient Budget".

## Time Limit (secs)

1

## Examples

Example 1

Input

12

4 3 3 6 5 2 8 8 2 7 8 2

2 1 1 2 3 6 1 6 2 7 6 3

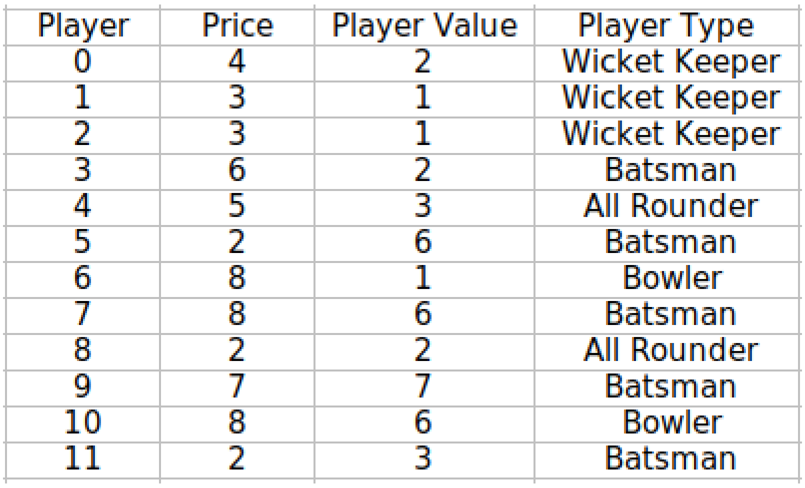
50

Output

34

Explanation

Consider the following table:

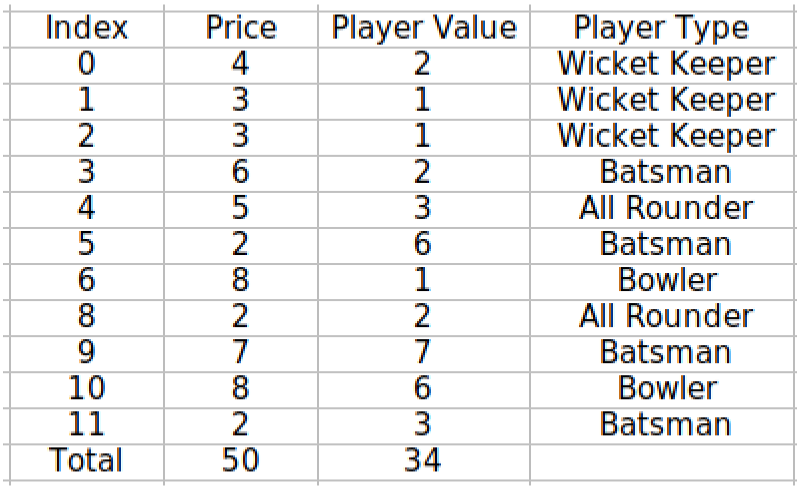


| Player | Price | Player Value | Player Type |
|---|---|---|---|
| 0 | 4 | 2 | Wicket Keeper |
| 1 | 3 | 1 | Wicket Keeper |
| 2 | 3 | 1 | Wicket Keeper |
| 3 | 6 | 2 | Batsman |
| 4 | 5 | 3 | All Rounder |
| 5 | 2 | 6 | Batsman |
| 6 | 8 | 1 | Bowler |
| 7 | 8 | 6 | Batsman |
| 8 | 2 | 2 | All Rounder |
| 9 | 7 | 7 | Batsman |
| 10 | 8 | 6 | Bowler |
| 11 | 2 | 3 | Batsman |

As can be seen from the table above, the player type has been calculated according to the index.

One needs to have a total of 11 players in the team, so we need to exclude 1 player and form the team. So, the player that is excluded is the player at index 7 with price 8 and value 6. So, the total max value that can be obtained is 34 with the total price of 50 which is within the budget while maintaining the constraint on player type viz. at least 2 batsmen and 2 bowlers and 1 wicket-keeper and 1 all-rounder.

Refer the below diagram for selected players.



| Index | Price | Player Value | Player Type |
|---|---|---|---|
| 0 | 4 | 2 | Wicket Keeper |
| 1 | 3 | 1 | Wicket Keeper |
| 2 | 3 | 1 | Wicket Keeper |
| 3 | 6 | 2 | Batsman |
| 4 | 5 | 3 | All Rounder |
| 5 | 2 | 6 | Batsman |
| 6 | 8 | 1 | Bowler |
| 8 | 2 | 2 | All Rounder |
| 9 | 7 | 7 | Batsman |
| 10 | 8 | 6 | Bowler |
| 11 | 2 | 3 | Batsman |
| Total | 50 | 34 | |

Example 2

Input

12

4 3 3 6 5 2 8 8 2 7 9 2

2 1 1 2 3 6 1 6 2 7 6 3

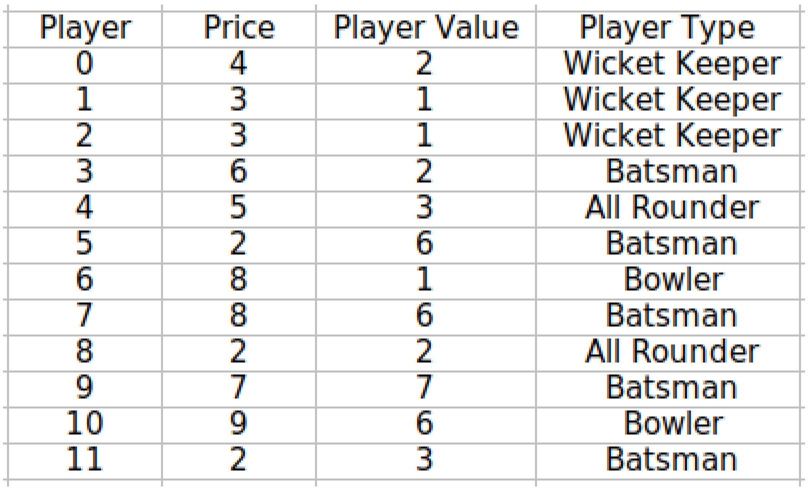
50

Output

Insufficient Budget

Explanation

Consider the following diagram:



| Player | Price | Player Value | Player Type |
|---|---|---|---|
| 0 | 4 | 2 | Wicket Keeper |
| 1 | 3 | 1 | Wicket Keeper |
| 2 | 3 | 1 | Wicket Keeper |
| 3 | 6 | 2 | Batsman |
| 4 | 5 | 3 | All Rounder |
| 5 | 2 | 6 | Batsman |
| 6 | 8 | 1 | Bowler |
| 7 | 8 | 6 | Batsman |
| 8 | 2 | 2 | All Rounder |
| 9 | 7 | 7 | Batsman |
| 10 | 9 | 6 | Bowler |
| 11 | 2 | 3 | Batsman |

As can be seen the player type has been allocated according to the index.

One needs to have a total of 11 players in the team where one needs to have at least 2 batsmen, 2 bowlers, 1 wicket-keeper and 1 all-rounder. To form such a team requires a minimum budget of 51 which violates the constraints on the budget. Hence, print "Insufficient Budget" as the output.