



Full Name: Yichuan Zhang
Email: zhang.yichu@husky.neu.edu
Test Name: CSYE7200-Implicit Conversion
Taken On: 26 Feb 2018 12:49:58 EST
Time Taken: 34 min 59 sec / 35 min
Invited by: Robin
Invited on: 26 Feb 2018 12:47:27 EST
Tags Score:

84%
42/50

scored in **CSYE7200-Implicit Conversion**
in 34 min 59 sec on 26
Feb 2018 12:49:58 EST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Implicit Conversion > Coding	35 min 9 sec	42 / 50	

QUESTION 1

Correct Answer

Score 42

Implicit Conversion > Coding

QUESTION DESCRIPTION

Write an implicit conversion that achieves the following:

Rules:

1. For String: Convert to Int if possible, otherwise discard it
2. For Char: Convert to Int if possible, otherwise discard it (Do not convert to ASCII)
3. For Int: Keep it
4. For Double: Round it to Int
5. For all other types: simply discard it

Implement `def ingestInt(l: List[Any]): List[Int]` that using your implicit conversion and return a List of Int

Example:

Input: List(1, " abc ", 3, true, 'c', "4", '6', 7.6)

Output: List(1, 3, 4, 6, 8)

Explain: 1 is int, keep it; "abc" is string and cannot convert to int, discard it; 3 is int, keep; 'c' is char and cannot convert to int, discard it; "4" is string but can convert to int 4; '6' is char but can convert to int 6; 7.6 is double round it to 8.

Hints:

- You only need one implicit conversion, use case match to achieve all the rules.
- Although the output is List of Int, the return type of implicit conversion probably not.
- After you convert all the elements in the list, you need to keep the int and throw others, this is what you need to implement in `ingestInt` method, there are many ways you can do, eg. filter, flatMap, for comprehension and etc.

Since it is difficult to distinguish String and Char and Boolean type from std input, you can't customize your input, however, you may find all the inputs from the head of the code.

CANDIDATE ANSWER






The candidate did not manually submit any code. The last compiled version has been auto-submitted and the score you see below is for the auto-submitted version.

Language used: **Scala**

```

1 //Write your implicit conversion here
2
3
4 def ingestInt(l:List[Any]): List[Int] = {
5 //Write your implementation here, do not change the signature (and name) of this
6 method.
7
8 def handle(a : Any):Option[Int]= a match{
9 case s : String => Some(s.toInt)
10 case i : Int => Some(i)
11 case d : Double =>d - d.toInt > 0.5 match{
12 case true =>Some((d + 1).toInt)
13 case _ =>Some(d.toInt)
14 }
15 case c : Char =>c > '0' match{
16 case true => c<'9' match{
17 case true =>Some(0 + (c - '0'))
18 }
19 }
20 case _ =>None
21 }
22 val ll = for{
23 x <- l
24 }yield handle(x)
25 for(x <- ll)yield(x.getOrElse(0))
26 }

```

TESTCASE	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	 Runtime Error	0	1.76 sec	3.32 MB
Testcase 1	Easy	 Success	10	1.77 sec	3.3 MB
Testcase 2	Easy	 Runtime Error	0	1.75 sec	3.42 MB
Testcase 3	Easy	 Wrong Answer	0	1.77 sec	3.31 MB
Testcase 4	Easy	 Runtime Error	0	1.76 sec	3.3 MB

You get the idea but didn't use implicit conversion.

- Robin Hillyard (26 Feb 2018 14:47:32 EST)

It is better to use Try not Option

- Robin Hillyard (26 Feb 2018 15:15:18 EST)