course_4_assessment_4

Due: 2019-02-04 15:16:00

Description: Assessment for the Exceptions lesson

Score: 8.0 of 8 = 100.0%

Questions

Score: 1.0 / 1

Comment: autograded

The code below takes the list of country, `country` , and searches to see if it is in the dictionary `gold` which shows some countries who won gold during the Olympics. However, this code currently does not work. Correctly add try/except clause in the code so that it will correctly populate the list, `country_gold` , with either the number of golds won or the string “Did not get gold”.

Save & Run

8/22/2020, 4:39:41 PM - 4 of 4

Show in CodeLens

```
gold = {"US": 46, "Fiji": 1, "Great Britain": 27, "Cuba": 5, "Thailand": 2, "China": 26, "
country = ["Fiji", "Chile", "Mexico", "France", "Norway", "US"]
country_gold = []
print(gold.keys())
for x in country:
    try:
        x in gold.keys()
        country_gold.append(gold[x])
    except KeyError:
        country_gold.append("Did not get gold")

print(country_gold)
```

```
['US', 'Fiji', 'Great Britain', 'Cuba', 'Thailand', 'China', 'France']
[1, 'Did not get gold', 'Did not get gold', 10, 'Did not get gold', 46]
```

ActiveCode (ac_exceptions_01)

Result	Actual Value	Expected Value	Notes
Pass	[1, '..., 46]	[1, '..., 46]	Testing that country_gold is assigned to correct values

Expand Differences

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

Provided is a buggy for loop that tries to accumulate some values out of some dictionaries. Insert a try/except so that the code passes.

Save & Run 8/22/2020, 4:40:32 PM - 2 of 2 Show in CodeLens

```
di = [{"Puppies": 17, 'Kittens': 9, "Birds": 23, 'Fish': 90, "Hamsters": 49},
      {"Puppies": 23, "Birds": 29, "Fish": 20, "Mice": 20, "Snakes": 7},
      {"Fish": 203, "Hamsters": 93, "Snakes": 25, "Kittens": 89},
      {"Birds": 20, "Puppies": 90, "Snakes": 21, "Fish": 10, "Kittens": 67}]
total = 0
for diction in di:
    try:
        diction.keys() == "Puppies"
        total = total + diction['Puppies']
    except:
        pass

print("Total number of puppies:", total)
```

```
Total number of puppies: 130
```

ActiveCode (ac_exceptions_011)

Result	Actual Value	Expected Value	Notes
Pass	130	130	Testing that total has the correct value.

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

The list, `numb` , contains integers. Write code that populates the list `remainder` with the remainder of 36 divided by each number in `numb` . For example, the first element should be 0, because 36/6 has no remainder. If there is an error, have the string “Error” appear in the `remainder` .

Save & Run 8/22/2020, 4:40:47 PM - 2 of 2 Show in CodeLens

```
numb = [6, 0, 36, 8, 2, 36, 0, 12, 60, 0, 45, 0, 3, 23]

remainder = []
for i in numb:
    if (i == 0):
```

```
        remainder.append("Error")
    elif (36 % i):
        remainder.append(36 % i)
    elif (36 % i == 0):
        remainder.append(0)
print(remainder)
```

```
[0, 'Error', 0, 4, 0, 0, 'Error', 0, 36, 'Error', 36, 'Error', 0, 13]
```

ActiveCode (ac_exceptions_02)

Result	Actual Value	Expected Value	Notes
Pass	[0, '..., 13]	[0, '..., 13]	Testing that remainder is assigned to correct values.

Expand Differences

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

Provided is buggy code, insert a try/except so that the code passes.

Save & Run 8/22/2020, 4:41:00 PM - 2 of 2 Show in CodeLens

```
lst = [2, 4, 10, 42, 12, 0, 4, 7, 21, 4, 83, 8, 5, 6, 8, 234, 5, 6, 523, 42, 34, 0, 234, 1

lst_three = []

for num in lst:
    try:
        if 3 % num == 0:
            lst_three.append(num)
    except ZeroDivisionError:
        pass
print(lst_three)
```

```
[1, 3]
```

ActiveCode (ac_exceptions_021)

Result	Actual Value	Expected Value	Notes
Pass	[1, 3]	[1, 3]	Testing that lst_three has the correct values.

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

Write code so that the buggy code provided works using a try/except. When the codes does not work in the try, have it append to the list `attempt` the string “Error”.

Save & Run 8/22/2020, 4:43:15 PM - 3 of 3 Show in CodeLens

```
full_lst = ["ab", 'cde', 'fgh', 'i', 'jkml', 'nop', 'qr', 's', 'tv', 'wxy', 'z']

attempt = []

for elem in full_lst:
    try:
        attempt.append(elem[1])
    except IndexError:
        attempt.append("Error")
print(attempt)
```

```
['b', 'd', 'g', 'Error', 'k', 'o', 'r', 'Error', 'v', 'x', 'Error']
```

ActiveCode (ac_exceptions_03)

Result	Actual Value	Expected Value	Notes
Pass	['b',...ror']	['b',...ror']	Testing that attempt has the correct values.

Expand Differences

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

The following code tries to append the third element of each list in `conts` to the new list `third_countries`. Currently, the code does not work. Add a try/except clause so the code runs without errors, and the string ‘Continent does not have 3 countries’ is appended to `countries` instead of producing an error.

Save & Run 8/22/2020, 4:41:28 PM - 2 of 2 Show in CodeLens

```
conts = [['Spain', 'France', 'Greece', 'Portugal', 'Romania', 'Germany'], ['USA', 'Mexico'
         ['Japan', 'China', 'Korea', 'Vietnam', 'Cambodia'],
         ['Argentina', 'Chile', 'Brazil', 'Ecuador', 'Uruguay', 'Venezuela'], ['Australia'
         ['Zimbabwe', 'Morocco', 'Kenya', 'Ethiopa', 'South Africa'], ['Antarctica']]

third_countries = []

for c in conts:
    try:
        third_countries.append(c[2])
    except IndexError:
        third_countries.append("Continent does not have 3 countries")
print(third_countries)
```

```
['Greece', 'Canada', 'Korea', 'Brazil', 'Continent does not have 3 countries', 'Kenya', 'Continent d
```

ActiveCode (ac_exceptions_031)

Result	Actual Value	Expected Value	Notes
Pass	['Gre...ies']	['Gre...ies']	Testing that third_countries is created correctly.

Expand Differences

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

The buggy code below prints out the value of the sport in the list `sport` . Use try/except so that the code will run properly. If the sport is not in the dictionary, `ppl_play` , add it in with the value of 1.

Save & Run 8/22/2020, 4:41:39 PM - 2 of 2 Show in CodeLens

```
sport = ["hockey", "basketball", "soccer", "tennis", "football", "baseball"]

ppl_play = {"hockey": 4, "soccer": 10, "football": 15, "tennis": 8}

for x in sport:
    try:
        print(ppl_play[x])
    except KeyError:
```

```
        ppl_play[x] = 1
```

```
4
10
8
15
```

ActiveCode (ac_exceptions_04)

Result	Actual Value	Expected Value	Notes
Pass	[('ba..., 8)]	[('ba..., 8)]	Testing that ppl_play is assigned to correct values.

Expand Differences

You passed: 100.0% of the tests

Score: 1.0 / 1

Comment: autograded

Provided is a buggy for loop that tries to accumulate some values out of some dictionaries. Insert a try/except so that the code passes. If the key is not there, initialize it in the dictionary and set the value to zero.

Save & Run 8/22/2020, 4:41:50 PM - 2 of 2 Show in CodeLens

```
di = [{"Puppies": 17, 'Kittens': 9, "Birds": 23, 'Fish': 90, "Hamsters": 49},
      {"Puppies": 23, "Birds": 29, "Fish": 20, "Mice": 20, "Snakes": 7},
      {"Fish": 203, "Hamsters": 93, "Snakes": 25, "Kittens": 89},
      {"Birds": 20, "Puppies": 90, "Snakes": 21, "Fish": 10, "Kittens": 67}]

total = 0
for diction in di:
    try:
        diction.keys() == "Puppies"
        total = total + diction['Puppies']
    except:
        pass
    if("Puppies" not in diction.keys()):
        diction["Puppies"] = 0
```

```
Total number of puppies: 130
```

ActiveCode (ac_exceptions_041)

Result	Actual Value	Expected Value	Notes
Pass	4	4	Testing that every dictionary in di has the key 'Puppies'.

You passed: 100.0% of the tests

Score Me