PA4

PA 4

CSE 8A Fall 2021 PA 4

Due date: Tuesday, Nov 2 @ 11:59PM PST

(No late submission is allowed)

Provided Files

None

File(s) to Submit

- `climb.py`
- `is_valid.py`

Part 1: Implementation

Background: You are climbing a staircase. Each time you can either climb `1` or `2` steps.

The staircase contains `45` steps. Suppose there are `m` different ways to climb to the step at index `n`. The index of first step is 0. The index of last step is 44.

Examples:

- when `n = 0,` then `m = 1`
  - Explanation: The only way is to climb 1 step to reach the step at index 0.

    ```
    1 step
    ```

- when `n = 1,` then `m = 2`
  - Explanation: There are two ways to climb to the step at index 1.

    ```
    1 step + 1 step
    ```

    ```
    2 steps
    ```

- when `n = 2,` then `m = 3`

- Explanation: There are three ways to climb to the step at index 2.

```
1 step + 1 step + 1 step
```

```
1 step + 2 steps
```

```
2 steps + 1 step
```

`climb.py`

`def valid_range(a,b)`

Task 0 (21 points): In `climb.py`, write a function `valid_range(a,b)` returning `True` or `False` to check whether `a` and `b` are specifying a valid **index** range of the staircase. We also specify that `a <= b`. Therefore, a valid range should satisfy the constraint

$$0 \leq a \leq b < 45$$

Once you have implemented this function, please come up with six (6) test cases of `valid_range(a,b)` and write them in `climb.py`.

A sample test case in your `climb.py` will look something like the following:

> `print("Expect: True,  got: ",valid_range(some_number, some_other_number))`

> `print("Expect: False, got: ",valid_range(some_number, some_other_number))`

- Three of your tests cases should return `True` and the rest should return `False`.
- Your test cases should be different than those provided in the auto-grader. There will not be hidden test cases for this portion or this PA overall.

Note that both the implementation of function `valid_range` and your test cases will be graded.

`def climb(a,b)`

Task 1 (52 points): (Again) In `climb.py`, write a function `climb(a,b)` which takes in two integers `a` and `b`, and returns a list where each element at index `i` is the total number of distinct ways to climb from the step at index `0` to the step at index `i`. For `climb(8,10)`, the elements of the result list look something like `[ways to get to step at index 8, ways to get to step at index 9, ways to get to step at index 10]`.

For example:

- `climb(0, 3)` will return `[1,2,3,5]`
  - Explanation: The first three elements `[1,2,3]` are explained in the background. The element at index 4, `5` represents the number of different ways to reach the fourth step. The five different ways are listed below:

```
1 step + 1 step + 1 step + 1 step
```

```
1 step + 2 steps + 1 step
```

```
1 step + 1 step + 2 steps
```

```
2 steps + 2 steps
```

```
2 steps + 1 step + 1 step
```

Call function `valid_range(a,b)` in function `climb(a,b)` to check whether `[a,b]` is a valid range. If `[a,b]` is not a valid range, function climb `(a,b)` should return an empty list (`[]`).

- More Examples:
  - If `a = 2` and `b = 4`, the output of function call `climb(a,b)` should be `[3,5,8]`.
    - The given staircase is steps 2, 3, 4. Therefore, the returning list is 3 ways to get to step 3, 5 ways to get to step 4, and 8 ways to get to step 5.
  - If `a = 4` and `b = 2`, the output of function call `climb(a,b)` should be `[]`.
  - If `a = 5` and `b = 5`, the output of function call `climb(a,b)` should be `[13]`.

`is_valid.py`

`def is_valid(num)`

Task 2 (12 points): In this task, we will suppose the staircase contains **infinite** number of steps. In `is_valid.py`, write a function `is_valid(num)` which takes in an integer `num` and returns `True` if `num` is the total number of ways to reach some step on the staircase. **(You will receive 0 points if you hardcode this task.)**

- To make your code more robust and efficient, remember to check whether the input parameter meet the requirements of the problem. In your function, you can check whether `num` is in the valid numeric range (think about what is a valid range for `num`) and handle the edge case(s) before doing any computation.
- Examples:
  - is_valid(-1) returns **False** since the input is negative
  - is_valid(1) returns **True** since there are 1 way to reach the step at index 0
  - is_valid(144) returns **True** since there are 144 ways to reach the step at index 10
  - ……

Part 2: Style (5 points)

Coding style is an important part of ensuring readability and maintainability of your code. We will grade your coding style in all submitted code files (**including optional extra credit portion if submitted**) according to the style guidelines. You can keep these guidelines in mind as you write your code or go back and fix your code at the end. For now, we will mainly be focusing on the

following:

- **Use of descriptive variable names**: The names of your variables should describe the data they hold. Your variable names should be words (or abbreviations), not single letters.
    - e.g. `a` --> `index_of_apple`; `letter1` and `letter2` --> `lower_case_letter` and `upper_case_letter`
    - Exception: If it is a loop index like `i`, `j`, `k`, one char is OK and sometimes preferred.
- **Inline Comments**: If there is a length of code that is left unexplained, take the time to type a non-redundant line summarizing this length of code (e.g. `#initialize an int` is redundant, vs. `#set initial length to 10 inches`). It will let others who look at your code understand what's going on without having to spend time understanding your logic first. But don't be too descriptive, as too many comments reduces readability.

Part 3: Conceptual Questions (10 points)

You are required to complete the Conceptual Questions in PA lesson. There is no time limit but you must submit by the PA deadline. You can submit multiple times before the deadline. Your latest submission will be graded.

Submission

Turning in your code

Submit all of the following files to PA Lesson on EdStem via the “Mark" Button by **Tuesday, Nov 2 @ 11:59PM PST** (**No late submission is allowed**) :

- `climb.py`
- `is_valid.py`

Evaluation

- **Correctness (85 points)** You will earn points based on the autograder tests that your code passes. There are hidden test cases for this PA. If the autograder tests are not able to run (e.g., your code does not compile or it does not match the specifications in this writeup), you may not earn credit. You latest submission will be graded.
- **Style (5 points)**
- **Conceptual Questions (10 points)**

PA 4 Conceptual Questions

Question 1 *Submitted Oct 27th 2021 at 5:51:54 pm*

What will be printed out by the following code?

```
def find_emotion(feeling, emotions):
    for idx in range(len(emotions)):
        if emotions[idx] == feeling:
            return True
        else:
            return False

emotions = ['joy', 'sadness', 'anger', 'fear', 'disgust']
print(find_emotion('fear', emotions))
```

- fear
- True
- **False**
- There is an error

Question 2 *Submitted Oct 27th 2021 at 5:59:01 pm*

Which of the following are equivalent to the following code? Select all that apply

```
print(1)
print(3)
print(5)
print(7)
print(9)
```

- [ ]
```
i = 1
while i<9:
  print(i)
  i += 2
```

- [ ]
```
i = 1
while i != 9:
  print(i)
  i += 2
```

- [ ]
```
i = 0
while i<5:
  print(2 * i + 1)
  i += 1
```

- [ ]
```
i = 1
while i!= 10:
  print(i)
  i += 2
```

- [ ]
```
for i in range(10, 2):
  print(i)
```

- [x]
```
for i in range(1, 10, 2):
  print(i)
```

Question 3 *Submitted Oct 27th 2021 at 5:58:53 pm*

What will happen when we run the following lines of code?

```
import random
words = ["eat", "drink", "sleep"]
words[random.randint(0, len(words))]
```

Reference for the randint function

`random.randint(a, b)`

Return a random integer *N* such that `a <= N <= b`. Alias for `randrange(a, b+1)`.

- Any of the three words will be printed always without any error
- Any of the three words will be printed but sometimes an error may be thrown
- Any of the first two words "eat" and "drink" will be printed always without any error
- Always the word "eat" will be printed always without any error

Question 4 *Submitted Oct 27th 2021 at 6:00:21 pm*

What will be printed by the following code

```
message = "hello"
for i in range(len(message)):
    if message[i] == 'l': #letter l
        break

    print(message[i], end = '')
```

- he
- hel
- hell
- hello
- this will cause an error

Question 5 *Submitted Oct 27th 2021 at 6:01:18 pm*

How many times does n change its value? Pleese count n = 4 as the first value change for n.

```
n = 4
x = 0
while n > 0:
    for i in range(n):
        x += 1
    n //= 2
print(x)
```

- 2
- 3
- 4
- 5
- 1
- 0