Quiz due Nov 23, 2022 07:30 +08

Problem 3

20/20 points (graded)

You are creating a song playlist for your next party. You have a collection of songs that can be represented as a list of tuples. Each tuple has the following elements:

- `name` : the first element, representing the song name (non-empty string)
- `song_length` : the second, element representing the song duration (float >= 0)
- `song_size` : the third, element representing the size on disk (float >= 0)

You want to try to optimize your playlist to play songs for as long as possible while making sure that the songs you pick do not take up more than a given amount of space on disk (the sizes should be less than or equal to the `max_disk_size`).

You decide the best way to achieve your goal is to start with the first song in the given song list. If the first song doesn't fit on disk, return an empty list. If there is enough space for this song, add it to the playlist.

For subsequent songs, you choose the next song such that its size on disk is smallest and that the song hasn't already been chosen. You do this until you cannot fit any more songs on the disk.

Write a function implementing this algorithm, that returns a **list of the song names in the order in which they were chosen, with the first element in the list being the song chosen first**. Assume song names are unique and all the songs have different sizes on disk and different durations.

You may not mutate any of the arguments.

For example,

- If `songs = [('Roar',4.4, 4.0),('Sail',3.5, 7.7),('Timber', 5.1, 6.9),('Wannabe',2.7, 1.2)]` and `max_size = 12.2` , the function will return `['Roar','Wannabe','Timber']`
- If `songs = [('Roar',4.4, 4.0),('Sail',3.5, 7.7),('Timber', 5.1, 6.9),('Wannabe',2.7, 1.2)]` and `max_size = 11` , the function will return `['Roar','Wannabe']`

```
def song_playlist(songs, max_size):
    """
    songs: list of tuples, ('song_name', song_len, song_size)
    max_size: float, maximum size of total songs that you can fit

    Start with the song first in the 'songs' list, then pick the next
    song to be the one with the lowest file size not already picked, repeat

    Returns: a list of a subset of songs fitting in 'max_size' in the order
             in which they were chosen.
    """
```

Paste your entire function (including the definition) in the box. Do not import anything. Do not leave any debugging print statements.

```
# Paste your code here
def song_playlist(songs, max_size):
    """
    songs: list of tuples, ('song_name', song_len, song_size)
    max_size: float, maximum size of total songs that you can fit
    Start with the song first in the 'songs' list, then pick the next
    song to be the one with the lowest file size not already picked, repeat
    Returns: a list of a subset of songs fitting in 'max_size' in the order
             in which they were chosen.
    """
    playlist = []
    if songs[0][2] <= max_size:
        playlist.append(songs[0][0])
    else:
        return playlist
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Test results

Hide output

CORRECT

Test: song_playlist([('a', 4.4, 4.0), ('b', 3.5, 7.7), ('c', 5.1, 6.9), ('d', 2.7, 1.2)], 20)

Output:

```
['a', 'd', 'c', 'b']
```

Test: song_playlist([('a', 4.4, 4.0), ('b', 3.5, 7.7), ('c', 5.1, 6.9), ('d', 2.7, 1.2)], 12.2)

Output:

```
['a', 'd', 'c']
```

Test: song_playlist([('a', 4.4, 4.0), ('b', 3.5, 7.7), ('c', 5.1, 6.9), ('d', 2.7, 1.2)], 11)

Output:

```
['a', 'd']
```

Test: song_playlist([('a', 4.0, 4.4), ('b', 7.7, 3.5), ('c', 6.9, 5.1), ('d', 1.2, 2.7)], 20)

Output:

```
['a', 'd', 'b', 'c']
```

Test: song_playlist([('a', 4.0, 4.4), ('b', 7.7, 3.5), ('c', 6.9, 5.1), ('d', 1.2, 2.7)], 12.3)

Output:

```
['a', 'd', 'b']
```

Test: song_playlist([('a', 1.4, 4.0)], 20)

Output:

```
['a']
```

Test: song_playlist([('aa', 4, 4), ('bb', 5, 7), ('cc', 5, 6), ('dd', 2, 1)], 1)

Output:

```
[]
```

Test: song_playlist([('aa', 4, 4), ('bb', 5, 7), ('cc', 5, 6), ('dd', 2, 1)], 3)

Output:

```
[]
```

Test: song_playlist([('z', 0.1, 0.1), ('a', 4.4, 4.0), ('b', 3.5, 7.7), ('c', 5.1, 6.9), ('d', 2.7, 1.2)], 1)

Output:

```
['z']
```

Test: song_playlist([('z', 0.1, 0.1), ('a', 4.4, 4.0), ('b', 3.5, 7.7), ('c', 5.1, 6.9), ('d', 2.7, 1.2)], 5.4)

Output:

```
['z', 'd', 'a']
```

Test: song_playlist([('z', 0.1, 0.1), ('a', 4.4, 4.0), ('b', 2.7, 1.2), ('cc', 3.5, 7.7), ('ddd', 5.1, 6.9)], 5.4)

Output:

```
['z', 'b', 'a']
```

Test: song_playlist([('z', 0.1, 9.1), ('a', 4.4, 4.0), ('b', 2.7, 1.2), ('cc', 3.5, 7.7), ('ddd', 5.1, 6.9)], 14)

Output:

```
['z', 'b']
```

Hide output

Hide output

Submit You have used 1 of 10 attempts

✔ Correct (20/20 points)