Bucketlist exercises

Exercise 1 : Golf stableford score calculator (Deadline Friday 23 February 23:59)

- A golf course has 18 holes. Each hole has an index from 1 to 18. Index 1 is the hardest hole and index 18 is the easiest hole.
- Each player has a handicap in the range [0-54]. The following are some examples of how handicaps works:
    - A player with a handicap of 6 receives “one free stroke” on hole indexes 1-6 (the six most difficult holes).
    - A player with a handicap of 18 receives “one free stroke” on every hole (indexes 1-18).
    - A player with a handicap of 25 receives “one free stroke” on indexes 1-18 and an additional “one free stroke” on indexes 1-7. Overall the player thus receives “two free strokes” on indexes 1-7 and “one free stroke” on indexes 8-18.
    - A player with a handicap of 36 receives “two free strokes” on every hole (indexes 1-18).
    - A player with a handicap of 40 receives “two free strokes” on every hole (indexes 1-18) and an additional free stroke on indexes 1-4. Overall the player thus receives “three free strokes” on indexes 1-4 and “two free strokes” on indexes 5-18.
    - A player with a handicap of 54 receives “three free strokes” on every hole (indexes 1-18).
    - etc.
- Every player takes some number of strokes on each hole. A player’s net strokes count is determined by subtracting the number of “free strokes” she receives on the hole from the number of strokes taken.
Score to par	Points
-7	9
-6	8
-5	7
-4	6
-3	5
-2	4
-1	3
0	2
1	1
2+	0

  A player’s score to par on each hole is determined by subtracting the hole’s par from the player’s net stroke count. Points are awarded as shown above. Each hole is a par 3, 4, or 5.
- **Handicap: 18**

Hole	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Index	16	4	8	2	18	14	10	6	12	11	15	5	13	1	17	7	3	9

Free	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

A player with a handicap of 18 receives a free stroke on every hole. Assume she takes 6 strokes on a par 4. Her net stroke count is 5. Her score to par is 5-4 = 1. Thus she is awarded 1 point. Assume the same player takes 5 strokes on a par 5. Her net stroke count is 4. Her score to par is 4-5 = -1. She is thus awarded 3 points. Assume the same player takes 2 strokes on a par 3. Her net stroke count is 1. Her score to par is 1-3 = -2. She is thus awarded 4 points.

- **Handicap: 0**

Hole	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Index	16	4	8	2	18	14	10	6	12	11	15	5	13	1	17	7	3	9
Free	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

A player with a handicap of 0 receives no free strokes on any hole. Assume she takes 4 strokes on a par 4. Her net stroke count is 4. Her score to par is 4-4 = 0. She is thus awarded 2 points.

- **Handicap: 20**

Hole	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Index	16	4	8	2	18	14	10	6	12	11	15	5	13	1	17	7	3	9
Free	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Free				1										1				

A player with a handicap of 20 receives two free strokes on hole indexes 1-2 and one free stroke on hole indexes 3-18. Assume she takes 5 strokes on the par 4 index 2. Her net stroke count is 3. Her score to par 3-4 = -1. She is thus awarded 3 points. Assume she takes 6 strokes on the par 5 index 15. Her net stroke count is 5. Her score to par is 5-5 = 0. She is thus awarded 2 points.

- A player's *stableford score* is the sum of all points scored over the 18 holes. Your task is to write a program called `stableford_122.py`. The program reads golf scores from `stdin` and prints a sorted list of results (in descending order of total stableford points). Lines read from `stdin` are ordered and structured as follows:

  hole_1_par hole_2_par ... hole_18_par (par for each hole)
  hole_1_index hole_2_index ... hole_18_index (index for each hole)
  player_1_name player_handicap strokes_hole_1 strokes_hole_2 ... strokes_hole_18
  player_2_name player_handicap strokes_hole_1 strokes_hole_2 ... strokes_hole_18
  player_3_name player_handicap strokes_hole_1 strokes_hole_2 ... strokes_hole_18
  ...
  player_N_name player_handicap strokes_hole_1 strokes_hole_2 ... strokes_hole_18

- A score of 'X' means that hole should be ignored for the calculation of the overall points for that player.

- A score that is neither numeric nor 'X' is illegal and the corresponding player is disqualified. When printing results disqualified players are listed last and in the order they are read from `stdin`.

- Here is an example:

```
$ cat golf_results.txt
4 5 3 4 4 5 3 5 3 5 3 4 4 4 4 5 4 3
16 4 8 2 18 14 10 6 12 11 15 5 13 1 17 7 3 9
Tweetie Pie 14 4 7 3 5 10 7 4 12 4 5 3 4 4 5 5 5 4 4
Penelope Pitstop 17 4 7 3 5 10 7 4 12 4 5 3 4 4 5 5 5 4 4
Bambi 19 5 6 4 5 5 6 4 6 ; 6 4 5 5 5 5 6 5 4
Sylvester The Cat 36 X X X X X X X X X X X X X X X X X X
Sneezy 18 5 6 4 5 5 6 4 6 4 6 4 5 5 5 5 6 5 4
Pinocchio 18 5 6 4 5 5 6 4 6 4 A 4 5 5 5 5 6 5 4
Shere Khan 7 4 X 3 4 4 5 1 4 3 5 3 X 4 4 4 5 4 3
Cruella de Vil 0 4 3 1 X 2 5 3 4 1 5 3 3 3 3 3 5 4 3

$ python3 stableford_122.py < golf_results.txt
   Cruella de Vil : 47
       Shere Khan : 40
 Penelope Pitstop : 38
           Sneezy : 36
      Tweetie Pie : 35
Sylvester The Cat :  0
             Bambi : Disqualified
         Pinocchio : Disqualified
```

- You can assume:
  - There will be no ties.
  - There is at least one player.
  - All final scores will be less than 100.
  - All handicaps are in the range 0-54 inclusive.
- If you are still struggling to understand how handicaps work here are two more examples:

Handicap: 9

Hole	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Index	16	4	8	2	18	14	10	6	12	11	15	5	13	1	17	7	3	9
Free		1	1	1				1				1		1		1	1	1

Handicap: 42

Hole	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Index	16	4	8	2	18	14	10	6	12	11	15	5	13	1	17	7	3	9
Free	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Free	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Free		1		1				1				1		1			1	

- If you are still struggling to understand how points are calculated here is a full worked example:

Handicap: 14

Hole	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Par	4	5	3	4	4	5	3	5	3	5	3	4	4	4	4	5	4	3
Index	16	4	8	2	18	14	10	6	12	11	15	5	13	1	17	7	3	9
Free		1	1	1		1	1	1	1	1		1	1	1		1	1	1
Stokes	4	7	3	5	10	7	4	12	4	5	3	4	4	5	5	5	4	4
Net	4	6	2	4	10	6	3	11	3	4	3	3	3	4	5	4	3	3
Points	2	1	3	2	0	1	2	0	2	3	2	3	3	2	1	3	3	2

Total points: 35

Exercise 2 : Longest common substring (Deadline Friday 23 March 23:59)

- Write a program called `lcsub_122.py` that reads two strings `s1` and `s2` from `stdin` (one per line). The program should print the longest substring common to both `s1` and `s2`, its length and the index at which it begins in `s2`. For example:

```
$ cat stdin.txt
tsobjrb3cq92v2brtb43r3e46yr8ys6776trbnywbo1jvkuxe0asqu9a
e2pe9znwft3vsd1w7q0b9bgzk2s300f7kxmr6ry9de776trbnyyzegle

$ python3 lcsub_122.py < stdin.txt
776trbny 8 42
```

- You can assume that `s1` and `s2` will always have a non-empty longest common substring and that it is unique.
- Note: The `s1` and `s2` strings need not be the same length and can each be up to ~3K characters in length.
- Note: There is a time-out on the Einstein tests which will cause inefficient solutions to fail.
- Note: Your solution may not import any modules other than the `sys` module.

Exercise 3 : Boggle (Deadline Monday 23 April 23:59)

- Boggle is a word game. Points are scored for creating words from adjacent letters in a 4x4 grid. Adjacent letters are next door to each other along a row, column or diagonal. When forming words each letter can be used only once.
- For each word formed a number of points are awarded. Points are awarded depending on the length of the word as follows:

Word length	Points
1, 2	0
3, 4	1
5	2
6	3
7	5
8+	11

- Here is an example boggle board:

a	t	e	e
a	p	y	o
t	i	n	u
e	d	s	e

- From the above board the following 29 scoring words can be formed where valid words are those listed in this dictionary for a total score of 33 points: ‘aid’, ‘die’, ‘end’, ‘ends’, ‘eye’, ‘one’, ‘ones’, ‘paid’, ‘pains’, ‘pat’, ‘pate’, ‘send’, ‘side’, ‘sin’, ‘sine’, ‘sit’, ‘site’, ‘tat’, ‘tie’, ‘tied’, ‘tin’, ‘tiny’, ‘type’, ‘unit’, ‘unite’, ‘united’, ‘use’, ‘yet’, ‘you’.

- From the same board the following 204 scoring words can be formed where valid words are those listed in this bigger dictionary for a total score of 281 points: ‘aid’, ‘aide’, ‘aids’, ‘ain’, ‘aine’, ‘ains’, ‘ais’, ‘ait’, ‘apaid’, ‘ape’, ‘apt’, ‘apted’, ‘atap’, ‘ate’, ‘dei’, ‘detain’, ‘detains’, ‘detinue’, ‘detinues’, ‘die’, ‘diet’, ‘din’, ‘dine’, ‘dines’, ‘dino’, ‘dins’, ‘dip’, ‘dipt’, ‘dis’, ‘dit’, ‘dita’, ‘dite’, ‘edit’, ‘eds’, ‘eine’, ‘end’, ‘endite’, ‘ends’, ‘ens’, ‘eon’, ‘eons’, ‘eta’, ‘etape’, ‘etat’, ‘eye’, ‘eyne’, ‘ide’, ‘ids’, ‘ins’, ‘ita’, ‘nid’, ‘nide’, ‘nids’, ‘nie’, ‘nied’, ‘nip’, ‘nipa’, ‘nis’, ‘nit’, ‘nite’, ‘nous’, ‘noy’, ‘nus’, ‘nye’, ‘one’, ‘ones’, ‘ons’, ‘onside’, ‘onus’, ‘oye’, ‘paid’, ‘pain’, ‘pains’, ‘pais’, ‘paise’, ‘pat’, ‘pate’, ‘pated’, ‘patin’, ‘patine’, ‘patines’, ‘patins’, ‘paty’, ‘pee’, ‘peeoy’, ‘peon’, ‘peones’, ‘peons’, ‘peony’, ‘pet’, ‘pia’, ‘pie’, ‘pied’, ‘piet’, ‘pieta’, ‘pin’, ‘pine’, ‘pines’, ‘pins’, ‘piny’, ‘pis’, ‘pise’, ‘pit’, ‘pita’, ‘pye’, ‘pyet’, ‘pyin’, ‘pyins’, ‘pyne’, ‘pynes’, ‘sdein’, ‘sen’, ‘send’, ‘side’, ‘sin’, ‘sind’, ‘sine’, ‘sip’, ‘sipe’, ‘sit’, ‘site’, ‘sited’, ‘snide’, ‘snip’, ‘snipe’, ‘snipy’, ‘snit’, ‘sny’, ‘snye’, ‘sue’, ‘sun’, ‘suni’, ‘tai’, ‘tain’, ‘tains’, ‘tais’, ‘tait’, ‘tap’, ‘tapa’, ‘tape’, ‘tapet’, ‘tapeta’, ‘tapis’, ‘tat’, ‘tate’, ‘tatie’, ‘tatin’, ‘ted’, ‘teds’, ‘tee’, ‘teind’, ‘teinds’, ‘tepa’, ‘tepid’, ‘tid’, ‘tide’, ‘tids’, ‘tie’, ‘tied’, ‘tin’, ‘tind’, ‘tinds’, ‘tine’, ‘tines’, ‘tins’, ‘tiny’, ‘tip’, ‘tipt’, ‘tis’, ‘tye’, ‘tyee’, ‘tyin’, ‘tynd’, ‘tynde’, ‘tyne’, ‘tynes’, ‘type’, ‘unde’, ‘uni’, ‘unis’, ‘unit’, ‘unite’, ‘united’, ‘unpaid’, ‘uns’, ‘use’, ‘yep’, ‘yet’, ‘yid’, ‘yids’, ‘yin’, ‘yins’, ‘yip’, ‘yipe’, ‘yipee’, ‘yite’, ‘yon’, ‘yond’, ‘yoni’, ‘yonis’, ‘you’, ‘yous’, ‘youse’, ‘yus’.

- Write a program called `boggle_122.py` that takes two arguments. The first argument is a text file containing a list of boggle boards (one per line with each represented as a single string of 16 letters e.g. the above board would be written ‘ateeapyotinuedse’). The second argument is a dictionary with one valid word per line. For each boggle board your program must print out the maximum number of points that can be scored using the provided dictionary.

- For example:

```
$ cat boards01.txt
ateeapyotinuedse
iddbetseshcgeago

$ python3 boggle_122.py boards01.txt dictionary02.txt
Possible points: 33
Possible points: 28

$ python3 boggle_122.py boards01.txt dictionary04.txt
Possible points: 281
Possible points: 330
```

- This is a challenging exercise that will require some research on your part. Not only must your program find and score all valid words but it must do so relatively *efficiently*. There is a timeout on the marker and if your program does not produce its answer before the timeout expires then you receive zero credit.

Frequently asked questions

- **Q.** My program works correctly with the provided sample input and produces the expected output. However, when I upload it to the checker it fails some test(s). Will you provide me with the input/output used by the checker?

A. No. The checker uses different sample input in order to test the robustness and generality of your solution. If you have built into your code dependencies on the specific sample input provided then your code is incorrect. It is up to you to define your own test cases to test the robustness and generality of your code. Developing suitable test cases is an integral part of the software engineering process. One approach is to start with the sample input provided with the exercise and then mutate it (e.g. if an exercise deals with names try different name lengths and formats) to develop additional test cases.

- **Q.** My program appears to work correctly but when I upload it the checker says it fails some test(s) and reports a `cmp: EOF` error. What's going on?
- **A.** A `cmp: EOF` error indicates the expected output and actual output have a differing number of lines. (Perhaps your program is outputting a blank line at the end where none is required.)

Marking

- To receive credit for completing an exercise your code must be uploaded here and pass all tests before the deadline.
- Marking:

Completed exercises	Marks
0	0
1	20
2	50
3	100