Assignment Guidelines.

- This assignment covers material in Module 06.
- Submission details:
  - Solutions to these questions must be placed in files `a06q1.py`, `a06q2.py`, `a06q3.py`, and `a06q4.py`, respectively, and must be completed using Python 3.
  - All solutions must be submitted to MarkUs. No solutions will be accepted through email, even if you are having issues with MarkUs.
  - Verify using MarkUs and your basic test results that your files were properly submitted and are readable on MarkUs.
  - For full style marks, your program must follow the Python section of the CS116 Style Guide.
  - Be sure to review the Academic Integrity policy on the Assignments page.
  - Helper functions need design recipe elements but not examples and tests.
- Download the testing module from the course web page. Include `import check` in each solution file.
- Restrictions:
  - Do not import any modules other than `math` and `check`.
  - Do not use any other Python functions not discussed in class or explicitly allowed elsewhere. See the allowable functions post on Piazza. You are always allowed to define your own helper functions, as long as they meet the assignment restrictions.
  - While you may use global `constants` in your solutions, do not use global `variables` for anything other than testing.
  - Read each question carefully for additional restrictions.
- The solutions you submit must be entirely your own work. Do not look up either full or partial solutions on the Internet or in printed sources.
Assignment 06
Due Wednesday, November 7 at 10:00 am (no late submissions)

Do NOT use recursion, map, or filter. All repetition must be performed using iteration (while and for loops only). Solutions using recursion will receive a correctness grade of 0.

Question 1:
Solve the following problems (a-e) from past assignments using loops. Do NOT use recursion, map, or filter. Solutions using recursion will receive a correctness grade of 0.
You don’t need to submit design recipe to any of the functions for this question (Question 1)

a) A03 Q2
b) A04 Q1b
c) A04 Q1c
d) A04 Q4
e) A05 Q2

Question 2:
The arctangent function can be calculated using the following summation:
\[
\sum_{i=0}^{\infty} (-1)^i \frac{x^{2i+1}}{2i+1} = (-1)^0 \frac{x^{2(0)+1}}{2(0)+1} + (-1)^1 \frac{x^{2(1)+1}}{2(1)+1} + (-1)^2 \frac{x^{2(2)+1}}{2(2)+1} + (-1)^3 \frac{x^{2(3)+1}}{2(3)+1} + (-1)^4 \frac{x^{2(4)+1}}{2(4)+1} + \cdots
\]

The function `arctan_eval` consumes a positive integer `n` and a floating point number `x` (between -1, 1) inclusive and returns an approximation of arctangent of `x` using the first `n` terms of the summation.

Example:
```
arctan_eval(4, 0.5) => 0.46346...
```
because
\[
(-1)^0 \frac{0.5^{2(0)+1}}{2(0)+1} + (-1)^1 \frac{0.5^{2(1)+1}}{2(1)+1} + (-1)^2 \frac{0.5^{2(2)+1}}{2(2)+1} + (-1)^3 \frac{0.5^{2(3)+1}}{2(3)+1} = 0.46346 \ldots
\]

Question 3:
In the provided interface for A06 you should find partial code to help you solve this problem.

In the interface file you will find the function `start` to create a random word for you and call the function `play` to initiate playing a game of guessing a word.

You will write the function `play` (header provided in the interface) that prompts the user to enter letters (one letter at a time) to guess the word (generated randomly by the function `start`), the game ends when the user guesses all the letters or when the user decides to terminate the game by entering “bye”. Check the following examples to see the exact steps how to design the game. (the user input shows in italic and bold)
Example 1:

```python
start()
welcome to Guess-The-Word Game
The word has 8 letters
To quit: type bye
Enter your guess - one letter: r
Bad guess
Enter your guess - one letter: l
Bad guess
Enter your guess - one letter: i
Good guess
Enter your guess - one letter: l
Bad guess
Enter your guess - one letter: p
Bad guess
Enter your guess - one letter: t
Good guess
Enter your guess - one letter: e
Bad guess
Enter your guess - one letter: c
Good guess
Enter your guess - one letter: o
Good guess
Enter your guess - one letter: n
Good guess
Enter your guess - one letter: d
Bad guess
Enter your guess - one letter: s
Bad guess
Enter your guess - one letter: r
Bad guess
Enter your guess - one letter: f
Good guess
Enter your guess - one letter: i
Good guess
Enter your guess - one letter: l
Bad guess
Enter your guess - one letter: u
Good guess
You won. Game over. The word was function
```
Example 2:
start()
welcome to Guess-The-Word Game
The word has 5 letters
To quit: type bye
Enter your guess - one letter: c
Good guess
Enter your guess - one letter: s
Good guess
Enter your guess - one letter: l
Good guess
Enter your guess - one letter: a
Good guess
You won. Game over. The word was class

Example 3:
start()
welcome to Guess-The-Word Game
The word has 6 letters
To quit: type bye
Enter your guess - one letter: r
Good guess
Enter your guess - one letter: y
Bad guess
Enter your guess - one letter: f
Bad guess
Enter your guess - one letter: g
Good guess
Enter your guess - one letter: bye

Note: You will write a complete design recipe to the function play and not start
Question 4:

Write a python function `print_fun` that consumes a positive natural number \( n \) less than 10 and prints a shape (of \( n \) rows and \( 2n \) columns) as shown in the examples.

Note: You are not allowed to reverse a string (i.e. `st[::-1]` or writing your own reverse function).

```python
>>> print_fun(1)
11
>>> print_fun(2)
2..2
2112
>>> print_fun(3)
3....3
32..23
321123
>>> print_fun(8)
8............8
87............78
876............678
8765............5678
87654............45678
876543............345678
8765432............2345678
8765432112345678
```