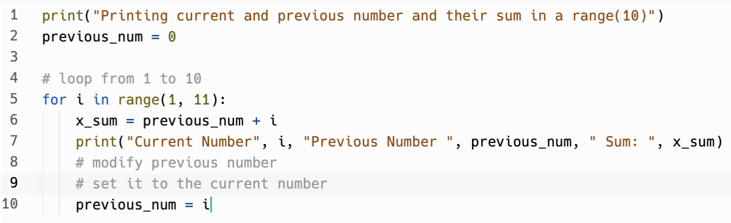
Name:
Student ID:
Signature:

Algorithms and Programming - II
YMT112 - Spring Semester, 2024

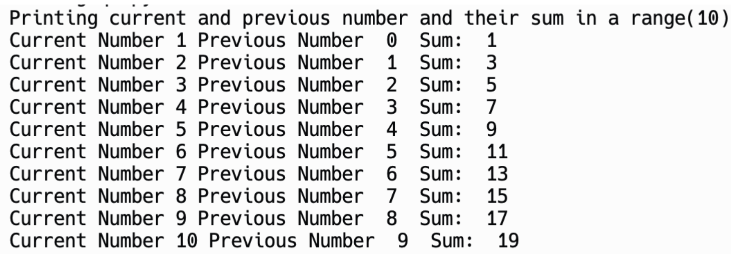
Wed Apr 17
Duration: 90 min.

1) **(15P) Write a Python program to compute the sum of the current number and its predecessor as you iterate through the first 10 numbers (from 1 to 10). The program should print the sum of the current number along with the previous number. A part of the expected output:**

```
Printing current and previous number and
their sum in a range(10)
Current Number 1 Previous Number 0 Sum: 1
...
Current Number 10 Previous Number 9 Sum: 19
```



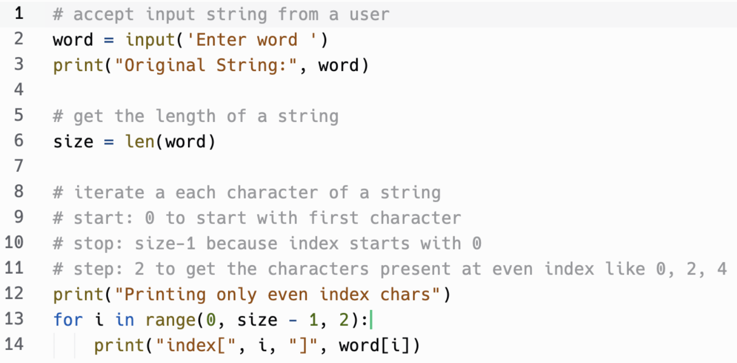
```python
print("Printing current and previous number and their sum in a range(10)")
previous_num = 0

# loop from 1 to 10
for i in range(1, 11):
    x_sum = previous_num + i
    print("Current Number", i, "Previous Number ", previous_num, " Sum: ", x_sum)
    # modify previous number
    # set it to the current number
    previous_num = i
```



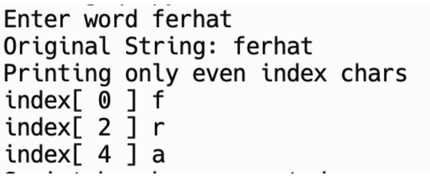
```
Printing current and previous number and their sum in a range(10)
Current Number 1 Previous Number  0  Sum:  1
Current Number 2 Previous Number  1  Sum:  3
Current Number 3 Previous Number  2  Sum:  5
Current Number 4 Previous Number  3  Sum:  7
Current Number 5 Previous Number  4  Sum:  9
Current Number 6 Previous Number  5  Sum:  11
Current Number 7 Previous Number  6  Sum:  13
Current Number 8 Previous Number  7  Sum:  15
Current Number 9 Previous Number  8  Sum:  17
Current Number 10 Previous Number  9  Sum:  19
```

2) **(15P) Write a program to accept a string from the user and display characters that are present at an even index number. Expected output:**

```
Orginal String is ferhat
Printing only even index chars
f
r
a
```



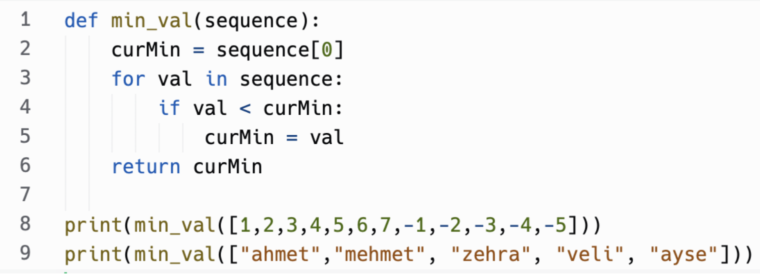
```python
# accept input string from a user
word = input('Enter word ')
print("Original String:", word)

# get the length of a string
size = len(word)

# iterate a each character of a string
# start: 0 to start with first character
# stop: size-1 because index starts with 0
# step: 2 to get the characters present at even index like 0, 2, 4
print("Printing only even index chars")
for i in range(0, size - 1, 2):
    print("index[", i, "]", word[i])
```



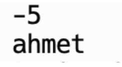
```
Enter word ferhat
Original String: ferhat
Printing only even index chars
index[ 0 ] f
index[ 2 ] r
index[ 4 ] a
```

3) **(15P) Write a function that returns the minimum value in a sequence.**

```
>> min_val([1,2,3,4,5,6,7,-1,-2,-3,-4,-5])
## -5
>> min_val(["buray", "ahmet", "ayşe", "hatice",
"okan", "zeynep"])
## "ahmet"
```



```python
def min_val(sequence):
    curMin = sequence[0]
    for val in sequence:
        if val < curMin:
            curMin = val
    return curMin

print(min_val([1,2,3,4,5,6,7,-1,-2,-3,-4,-5]))
print(min_val(["ahmet","mehmet", "zehra", "veli", "ayse"]))
```



```
-5
ahmet
```

4) **Print the results of those code snippets, please type the answer next to every snippet:**

a) (2.5P)

```
result = 0
for element in range(5):                # [0,1,2,3,4]
   result = result + element
print("The sum is: " + str(result))
```

The sum is: 10

b) (2.5P)

```
result = 0
for element in range(5,1,-1):
   result = result + element
print("The sum is: " + str(result))
```

5, 4, 3, 2
The sum is: 14

c) (2.5P)

```
result = 0
for element in range(0,8,2):
   result = result + element
print("The sum is: " + str(result))
```

0, 2, 4, 6
The sum is: 12

d)(2.5P)

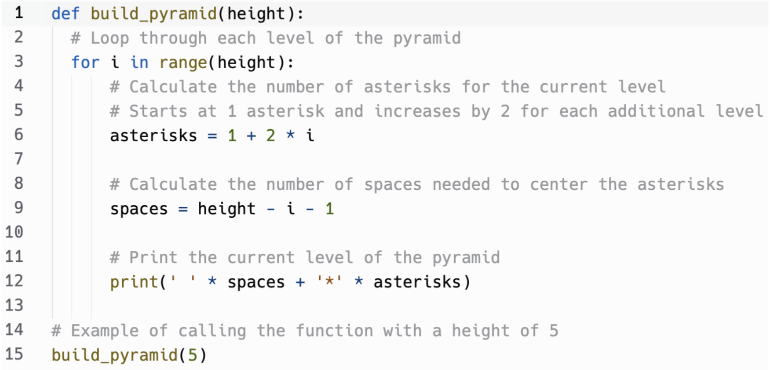
```
result = 0
size = 5
for element in range(size):
   result = result + element
print("When size = " + str(size) + ", the
result is " + str(result))
```

0, 1, 2, 3, 4
When size = 5, the result is 10

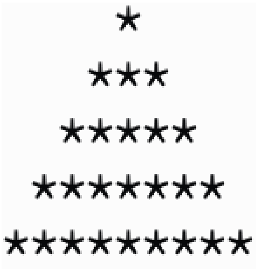
5)

(25P) Write a Python function named `build_pyramid` that generates a visual representation of a pyramid using asterisks $(*)$. The function should take a single integer parameter, height, which represents the number of levels in the pyramid. Each level should consist of a row of asterisks, and the number of asterisks should increase by two with each level, starting from one asterisk at the top. The pyramid should be centered on the widest row at the bottom.
For a call like `build_pyramid(3)`, the output is:

```
  *
 * * *
* * * * *
```



```python
def build_pyramid(height):
    # Loop through each level of the pyramid
    for i in range(height):
        # Calculate the number of asterisks for the current level
        # Starts at 1 asterisk and increases by 2 for each additional level
        asterisks = 1 + 2 * i

        # Calculate the number of spaces needed to center the asterisks
        spaces = height - i - 1

        # Print the current level of the pyramid
        print(' ' * spaces + '*' * asterisks)

# Example of calling the function with a height of 5
build_pyramid(5)
```

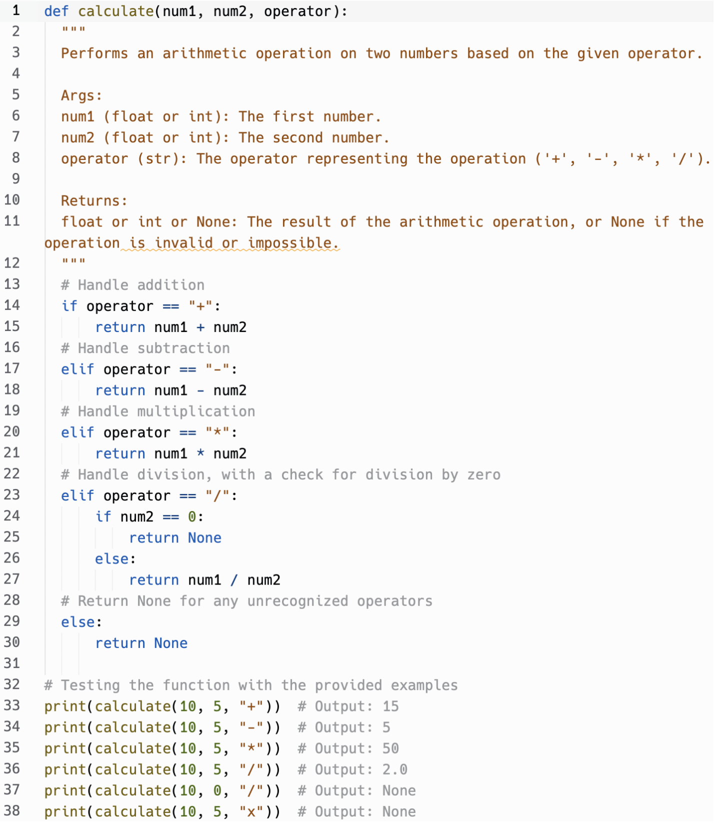


```
    *
   ***
  *****
 *******
*********
```

6)

(20P) Write a Python function named `calculate`

It performs arithmetic operations based on two numbers and an operator passed as arguments. The function should handle addition, subtraction, multiplication, and division.
It takes three arguments:
-a number
-another number
-an operator $(+, -, *, /)$ and returns the result of the operation.
`calculate(10, 5, "*")` should return `50`.
`calculate(10, 0, "/")` should return `None`.
`calculate(10, 5, "x")` should return `None`.



```python
def calculate(num1, num2, operator):
    """
    Performs an arithmetic operation on two numbers based on the given operator.

    Args:
    num1 (float or int): The first number.
    num2 (float or int): The second number.
    operator (str): The operator representing the operation ('+', '-', '*', '/').

    Returns:
    float or int or None: The result of the arithmetic operation, or None if the operation is invalid or impossible.
    """
    # Handle addition
    if operator == "+":
        return num1 + num2
    # Handle subtraction
    elif operator == "-":
        return num1 - num2
    # Handle multiplication
    elif operator == "*":
        return num1 * num2
    # Handle division, with a check for division by zero
    elif operator == "/":
        if num2 == 0:
            return None
        else:
            return num1 / num2
    # Return None for any unrecognized operators
    else:
        return None

# Testing the function with the provided examples
print(calculate(10, 5, "+"))  # Output: 15
print(calculate(10, 5, "-"))  # Output: 5
print(calculate(10, 5, "*"))  # Output: 50
print(calculate(10, 5, "/"))  # Output: 2.0
print(calculate(10, 0, "/"))  # Output: None
print(calculate(10, 5, "x"))  # Output: None
```