Assignment_1

February 16, 2024

1 Assignment 1

Peder Ørmen Bukaasen, Bård Tollef Pedersen and Eivind Lid Trøen

1.1 Exercise 1

The base case is when the list has only one number, then return that number. If it has more than one number, add the first element to the sum of the rest of the list. The sum of the rest of the list is calculated by calling the function with the rest of the list as an argument. This is done until the base case is reached.

So for the list [1, 2, 3, 4], the function will return 1 + sum_of_numbers([2, 3, 4]). Then it will return 2 + sum_of_numbers([3, 4]), then 3 + sum_of_numbers([4]), and finally 4. Backtracking, it will return 4 + 3, then 7 + 2, and finally 9 + 1, which is 10.

[ ]:
```
def sum_of_numbers(n_list):
    if len(n_list) == 1:
        return n_list[0]
    return n_list[0] + sum_of_numbers(n_list[1:])

list_2 = [1, 2, 3, 4]
print("sum of list: ", sum_of_numbers(list_2))  # 10
```

```
sum of list:  10
```

1.2 Exercise 2

Python class named Rectangle that represents a rectangle. This class include:

A constructor that takes the length and width of the rectangle as inputs. A method area() that computes and returns the area of the rectangle. A method perimeter() that computes and returns the perimeter of the rectangle. The class has appropriate checks to ensure that the length and width are positive numbers.

[ ]:
```
class Rectangle:
    def __init__(self, length, width):
        if length <= 0 or width <= 0:
            raise ValueError("Length and width must be positive numbers, larger␣
 ↪than 0")
        self.length = length
```

```
        self.width = width

    def area(self):
        return self.length * self.width

    def perimeter(self):
        return 2 * (self.length + self.width)

rectangle = Rectangle(5, 4)
print("Area of rectangle: ", rectangle.area()) # 20
print("Perimeter of rectangle: ", rectangle.perimeter()) # 18
```

```
Area of rectangle:  20
Perimeter of rectangle:  18
```

1.3 Exercise 3

The base case is when the list has only one number, then return that number. If it has more than one number, compare the first element with the minimum of the rest of the list. The minimum of the rest of the list is calculated by calling the function with the rest of the list as an argument. This is done until the base case is reached.

So for the list [7, 2, 5, 1, 8], the function will return the minimum of 7 and minimum([2, 5, 1, 8]). Then it will return the minimum of 2 and minimum([5, 1, 8]), then 1 and minimum([5, 8]), and finally 5. Backtracking, it will return check smallest of 5 and 1, then 1 and 2, and finally 1 and 7, which is 1.

[ ]:
```
def minimum(list):
    if len(list) == 1:
        return list[0]

    if list[0] < minimum(list[1:]):
        return list[0]

    return minimum(list[1:])

list_1 = [7, 2, 5, 1, 8]
print("The minimum value of list: ",minimum(list_1)) # 1
```

```
The minimum value of list:  1
```

[ ]:
```

```