TASK 1: Write a python program use the file scores.txt

(a) Find the average of the points scored over all the games in the file.

(b) Pick your favorite team and scan through file to determine how many games they won and how many games they lost.

(c) Find the team(s) that lost by 30 or more points the most times

(d) Find all the teams that averaged at least 70 points a game.

(e) Find all the teams that had winning records but were collectively outscored by their opponents. A team is collectively outscored by their opponents if the total number of points the team scored over all their games is less than the total number of points their opponents scored in their games against the team

example for task e

Game	Team 1	Score 1	Team 2	Score 2
1	A	70	B	85
2	A	80	C	85
3	B	100	C	80
4	C	60	A	50

Team A:

- **Total Points Scored:** 70 + 80 + 50 = 200 **Total Points Against:** 85 + 85 + 60 = 230 **Wins:** 0 **Losses:** 3

Team B:

- **Total Points Scored:** 85 + 100 = 185 **Total Points Against:** 70 + 80 = 150 **Wins:** 2 **Losses:** 0

Team C:

- **Total Points Scored:** 85 + 80 + 60 = 225 **Total Points Against:** 80 + 100 + 50 = 230 **Wins:** 2 **Losses:** 1

Conclusion:

In this scenario, **Team C** satisfies the condition.

-They have a winning record (2-1) and (wins>Losses)

- **Total Points Scored** < **Total Points Against** (outscored condition satisfy)

Task 2 You own a pizzeria named Olly's Pizzas and want to create a Python program to handle the customers and revenue.

Create the following classes with the following methods:

Class Pizza containing 1. **init method:** to initialize the size (small, medium, large), toppings (corn, tomato, onion, capsicum, mushroom, olives, broccoli), cheese (mozzarella, feta, cheddar). Note: One pizza can have only one size but many toppings and cheese. (1.5 marks) Throw custom exceptions if the selects toppings or cheese not available in lists given above. (1 mark) 2. **Price method**: to calculate the prize of the pizza in the following way: - small = 50, medium = 100, large= 200 Each topping costs 20 rupees extra, except broccoli, olives and mushroom, which are exotic and socost 50 rupees each. Each type of cheese costs an extra 50 rupees. (1.5 marks)

Class Order containing

1. **init method**: to initialize the name, customerid of the customer who placed the order (0.5 marks)

2. **order method**: to allow the customer to select pizzas with choice of toppings and cheese (1 mark)

3. **bill method:** to generate details about each pizza ordered by the customer and the total cost of the order. (2 marks)
*Note: A customer can get multiple pizzas in one order.

```
try:
    a="6"
    b=7
    c=0
    print(b+=c)
except TypeError:
    print(1)
except ZeroDivisionError:
    print(0)
except NameError:
    print("name")
except Exception as e:
    print("h")
```

```
try:
    a="6"
    b=7
    c=0
    print(d+b/c)
except TypeError:
    print(1)
except ZeroDivisionError:
    print(0)
except NameError:
    print("name")
except Exception as e:
    print("h")
```

```
try:
    a="6"
    b=7
    c=0
    print(a+b/c)
except TypeError:
    print(1)
except ZeroDivisionError:
    print(0)
except NameError:
    print("name")
except Exception as e:
    print("h")
```

```
class A:
    a=9
    def v(self,num):
        self.a+=num
obj=A()
obj.v(10)
obj1=A())
print(obj1.a)
```

```
class A:
    a=9
    def v(self,num):
        self.a+=num
obj=A()
obj.a+=9
obj.v(10)
print(obj.a)
```

```
def f():
    try:
        print(1)
        return 3
    return "a"
    except:
        print(2)
        return 6
print(f())
```

```
def f():
    try:
        print(1)
        return 3

    finally:
        print(2)
        return 6
print(f())
```

```
def f():
    try:
        print(1)
        return 3

    except:
        print(2)
        return 6
print(f())
```

```
try:
    a="5.5"
    print(a)
    a=int(a)
    print(b+a)
except NameError:
    print("name")
finally:
    print("All the best")
```

```
class A:
    a=9
    def v(self,num):
        self.a+=num
        A.a+=self.a
obj=A()
obj.a=15
obj.v(10)
obj1=A()
print(obj1.a)
```

```
class A:
    a=9
    def __init__(self):
        self.a+=25
    def v(self,num):
        self.a+=num
        A.a+=self.a
obj=A()
obj.a=15
obj.v(10)
obj1=A()
print(obj1.a)
```