Name: Dhruvi Virani

GID: 158410

# STRATEGY 2:

S1\_vm = {"S1":7,"S2":6,"S3":5,"S4":9}  
S1\_bid = {"S1":20,"S2":23,"S3":19,"S4":30}  
U1\_vm={"B1":4 ,"B2":3,"B3":2,"B4":5 }  
U1\_bid={"B1":25 ,"B2":22,"B3":17,"B4":21 }  
sorted\_s1 = dict(sorted(S1\_bid.items(), key=lambda kv: kv[1]))  
sorted\_u1 = dict(sorted(U1\_bid.items(), key=lambda kv: kv[1]))  
  
dict1={}  
for i in sorted\_u1.keys():  
 for j in sorted\_s1.keys():  
 if S1\_vm[j]>=U1\_vm[i]:  
 dict1[i]=j  
 S1\_vm[j]=S1\_vm[j]-U1\_vm[i]  
 #del U1\_vm[i]  
 break  
print("Allocation: ",dict1)  
  
lst\_seller=[]  
lst\_user=[]  
for i,j in dict1.items():  
 lst\_seller.append(S1\_bid[j])  
 lst\_user.append(U1\_bid[i])  
print("Price to be paid by each buyer will be: ",min(lst\_seller))  
print("Payment received to all sellers will be: ",max(lst\_user))  
  
a1={}  
a2={}  
for i,j in dict1.items():  
 p=S1\_bid[j]-min(lst\_seller)  
 q=max(lst\_user)-U1\_bid[i]  
 a1[j]=p  
 a2[i]=q  
print("loss of each seller is ",a1)  
print("loss of each user is ",a2)  
  
sum1=0  
for i, j in a1.items():  
 for k, m in dict1.items():  
 if (i==m):  
 sum1=sum1+(j\*U1\_vm[k])  
print("the loss of user per each machine: ",sum1)  
print("the average loss of seller after selling the machines: ",sum1/len(a1))  
  
sum2=0  
for i,j in a2.items():  
 for k,m in U1\_vm.items():  
 if(i==k):  
 sum2=sum2 + (j\*m)  
print("the loss of user per each machine: ",sum2)  
print("the average loss of seller after selling the machines: ",sum2/len(a2))  
  
b1=len(a1)  
b2=len(a2)  
print("the average loss of buyer per machine is ",(sum(a1.values())/b1))  
print("the average loss of user per machine is ",(sum(a2.values())/b2))

In this strategy 2, it is similar to Strategy 1 but here we are arranging sellers and buyers in ascending order.

The data is:

S1\_vm = {"S1":7,"S2":6,"S3":5,"S4":9}  
S1\_bid = {"S1":20,"S2":23,"S3":19,"S4":30}  
U1\_vm={"B1":4 ,"B2":3,"B3":2,"B4":5 }  
U1\_bid={"B1":25 ,"B2":22,"B3":17,"B4":21 }

# STRATEGY 3:

S1\_vm = {"S1":7,"S2":6,"S3":5,"S4":9}  
S1\_bid = {"S1":20,"S2":23,"S3":19,"S4":24}  
U1\_vm={"B1":4 ,"B2":3,"B3":5,"B4":9 }  
U1\_bid={"B1":25 ,"B2":22,"B3":17,"B4":21 }  
  
sorted\_s1 = dict(sorted(S1\_bid.items(), key=lambda kv: kv[1]))  
sorted\_u1 = dict(sorted(U1\_bid.items(), key=lambda kv: kv[1], reverse=True))  
print(sorted\_s1)  
print(sorted\_u1)  
dict1={}  
for i in sorted\_u1.keys():  
 for j in sorted\_s1.keys():  
 if S1\_vm[j]>=U1\_vm[i]:  
 dict1[i]=j  
 S1\_vm[j]=S1\_vm[j]-U1\_vm[i]  
 #del U1\_vm[i]  
 break  
print("Allocation: ",dict1)  
  
#buyer\_bid[i]>seller\_bid[i]  
seller\_break={}  
user\_break={}  
for i,j in dict1.items():  
 if U1\_bid[i]>S1\_bid[j]:  
 seller\_break=j  
 user\_break=i  
print(seller\_break)  
print(user\_break)  
  
g1={}  
h1={}  
for i,j in dict1.items():  
 if U1\_bid[i]<S1\_bid[j]:  
 g=S1\_bid[seller\_break]-S1\_bid[j]  
 h=U1\_bid[user\_break]-U1\_bid[i]  
 g1[g]=g  
 h1[h]=h  
print(g)  
print(h)  
  
lst\_seller=[]  
lst\_user=[]  
for i,j in dict1.items():  
 lst\_seller.append(S1\_bid[j])  
 lst\_user.append(U1\_bid[i])  
print("Price to be paid by each buyer will be: ",max(lst\_seller))  
print("Payment received to all sellers will be: ",max(lst\_user))  
  
a1={}  
a2={}  
for i,j in dict1.items():  
 p=S1\_bid[seller\_break]-S1\_bid[j]  
 q=U1\_bid[user\_break]-U1\_bid[i]  
 a1[j]=p  
 a2[i]=q  
print("loss of each seller is ",a1)  
print("loss of each user is ",a2)  
  
sum1=0  
for i, j in a1.items():  
 for k, m in dict1.items():  
 if (i==m):  
 sum1=sum1+(j\*U1\_vm[k])  
print("the loss of seller per each machine: ",sum1)  
print("the average loss of seller after selling the machines: ",sum1/len(a1))  
  
sum2=0  
for i,j in a2.items():  
 for k,m in U1\_vm.items():  
 if(i==k):  
 sum2=sum2 + (j\*m)  
print("the loss of user per each machine: ",sum2)  
print("the average loss of buyer after selling the machines: ",sum2/len(a2))

The data is:

S1\_vm = {"S1":7,"S2":6,"S3":5,"S4":9}  
S1\_bid = {"S1":20,"S2":23,"S3":19,"S4":24}  
U1\_vm={"B1":4 ,"B2":3,"B3":5,"B4":9 }  
U1\_bid={"B1":25 ,"B2":22,"B3":17,"B4":21 }

Steps:

Sort sellers in ascending order

○ Sort buyers in descending order

○ Allocate if required VMs are available

○ Find a break-even point: Break-even point is the largest index i for which buyer\_bid[i]>seller\_bid[i]

○ For example in given figure all Si are buyers and all ri are sellers. So here 3rd index is break-even point.

Price to paid by all buyers= bid of the buyer at break-even point

● Price to be paid to all sellers=bid of the seller at break-even point

In this strategy, we are finding break point of seller by checking the bids of buyers and sellers.

We check that if the bid of seller is greater than the allocated buyer, then it will check another bid till it will find the bid of seller is less than buyer. If it is less than buyer’s bid, it will consider the bid value of latest index.