```
#other_side:[]                    original location:[90,80,60,40,20] boat
#other_side:[60,40]boat           original location:[90,80,20]
#other_side:[60]                  original location:[90,80,40,20]boat
#other_side:[60,90]boat           original location:[80,40,20]
#other_side:[90]                  original location:[80,60,40,20]boat
#other_side:[90,60,40] boat       original location:[80,20]
#other_side:[90,60]               original location:[80,40,20]boat
#other_side:[90,60,80,20]boat     original location:[40]
#other_side:[90,80,20]            original location:[60,40]boat
#other_side:[90,80,60,40,20]boat  original location:[]

class Passengers:

    def __init__(self,pasenger,weight,animate):
        self.pasenger = pasenger
        self.weight = weight
        self.animate = animate
        self.index_other = set()
        self.boat_cap = 100

    def orginal_location(self):



        paired = False

        for person in range(len(self.weight)):
            for person2 in range(person + 1,len(self.weight)):
                if person not in self.index_other and person2 not in self.index_other:
                    if self.weight[person] + self.weight[person2] <= self.boat_cap:
                        print(f'{self.pasenger[person]} and {self.pasenger[person2]} traveling to safe island')
                        self.index_other.add(person)
                        self.index_other.add(person2)

                        self.update_notif()
                        paired = True
                        break

            if paired:
                break


        #dapat mag isa sasakay sa boat if walang ka pair at kapag mabigat yung weight
        if not paired:
            for i in range(len(self.weight)):
                if i not in self.index_other:
                    if self.weight[i] > 60:
                        print(f'{self.pasenger[i]} is traveling alone to safe')
                        self.index_other.add(i)
                        self.update_notif()
                        break


        self.other_side()

    def update_notif(self):
        Remaining = [self.pasenger[i] for i in range(len(self.pasenger)) if i not in self.index_other]
        Safe = [self.pasenger[i] for i in range(len(self.pasenger)) if i in self.index_other]
        print(f'Safe: {Safe}')
        print(f'Remaining: {Remaining}')
        print('\n')

    def other_side(self):
    # DAPAT BUMALIK YUNG BOAT SA ISLAND KAPAG YUNG LEN NG SELF.INDEX_OTHER HINDI PANTAY SA LEN NG SELF.PASSENGER
        saved_index = None



        if len(self.index_other) != len(self.pasenger):

            for index in self.index_other:
                if (self.weight[index] == 40 or self.weight[index] == 60)  and self.animate[index] != 1: #DISCARD YUNG INDEX NG PINAKAMAGAAN
                    print(f'{self.pasenger[index]} is traveling back ')


                    saved_index = index
```

```
                    break

#           print(saved_index)

            self.index_other.discard(saved_index)
#           print(self.index_other)

            self.update_notif()
            self.orginal_location()


        else:
           print("all pasenger travel successfully")

person = ["Supply","Robin","Llyod","Verlin","Roman"]
weights = [20,40,60,80,90]
animate = [1,0,0,0,0]



pogi = Passengers(person,weights,animate)

pogi.orginal_location()
```

```
Supply and Robin traveling to safe island
Safe: ['Supply', 'Robin']
Remaining: ['Llyod', 'Verlin', 'Roman']


Robin is traveling back
Safe: ['Supply']
Remaining: ['Robin', 'Llyod', 'Verlin', 'Roman']


Robin and Llyod traveling to safe island
Safe: ['Supply', 'Robin', 'Llyod']
Remaining: ['Verlin', 'Roman']


Robin is traveling back
Safe: ['Supply', 'Llyod']
Remaining: ['Robin', 'Verlin', 'Roman']


Verlin is traveling alone to safe
Safe: ['Supply', 'Llyod', 'Verlin']
Remaining: ['Robin', 'Roman']


Llyod is traveling back
Safe: ['Supply', 'Verlin']
Remaining: ['Robin', 'Llyod', 'Roman']


Robin and Llyod traveling to safe island
Safe: ['Supply', 'Robin', 'Llyod', 'Verlin']
Remaining: ['Roman']


Robin is traveling back
Safe: ['Supply', 'Llyod', 'Verlin']
Remaining: ['Robin', 'Roman']


Roman is traveling alone to safe
Safe: ['Supply', 'Llyod', 'Verlin', 'Roman']
Remaining: ['Robin']


Llyod is traveling back
Safe: ['Supply', 'Verlin', 'Roman']
Remaining: ['Robin', 'Llyod']


Robin and Llyod traveling to safe island
Safe: ['Supply', 'Robin', 'Llyod', 'Verlin', 'Roman']
Remaining: []


all pasenger travel successfully
```