32.Calorie Requiremenmt Calculation

```
def calculate_calories(s1,s2):
    if s1[0].lower()=='male':
        if s1[1].lower()=='sedentary':
            a=1.2
        elif s1[1].lower()=='moderately active':
            a=1.55
        else:
            a=1.9
        c=((10*s2[2])+(6.25*s2[1])-(5*s2[0])-161)*a
        print(f"To maintain your current weight,you'll need {c:.1f} calories per day.")
    else:
        if s1[0].lower()=='female':
            if s1[1].lower()=='sedentary':
                a=1.2
            elif s1[1].lower()=='moderately active':
                a=1.55
            else:
                a=1.9
            c=((10*s2[2])+(6.25*s2[1])-(5*s2[0])+5)*a
            print(f"To maintain your current weight,you'll need {c:.1f} calories per day.")
s1=list(map(str,input("Enter the gender and activity level(as comma-seperatedvalues)\n").split(",")))
s2=list(map(int,input("Enter the age,height and weight level(as comma-seperatedvalues)\n").split(",")))
calculate_calories(s1,s2)
```

```
Enter the gender and activity level(as comma-seperatedvalues)
female,sedentary
Enter the age,height and weight level(as comma-seperatedvalues)
23,153,60
Based on your inputs, you need 1735.5 calories per day.
```

31.Daily Temperature

```
def find_average_temperature(s1):
    b=list(map(int,s1))
    print("The average temperature is",sum(b)/len(b))
    print("The day with the highest temperatue is",(b.index(max(b))+1))

s1=list(map(str,input("Enter the temperatue of each day,seperated by commas:\n").split(",")))
find_average_temperature(s1)
```

```
Enter the temperatue of each day,seperated by commas:
31,33,35,37,28,33,41,36
The average temperature is 34.25
The day with the highest temperatue is 7
```

29.Movie Ratings

```
def check_rating(v):
    z=[]
    s=[]
    for i in v:
        if(i>0 and i<=5):
            z.append(i)
        elif (i>=6 and i<=10):
            s.append(i)
        else:
            print("Invalid input")
    if len(z)>len(s):
        print("The highest rating is for 0-5")
    else:
        print("The highest rating is for 6-10")

n=int(input("Enter the number of viewers:\n"))
v=[]
for i in range(n):
    k=int(input())
    v.append(k)
check_rating(v)
```

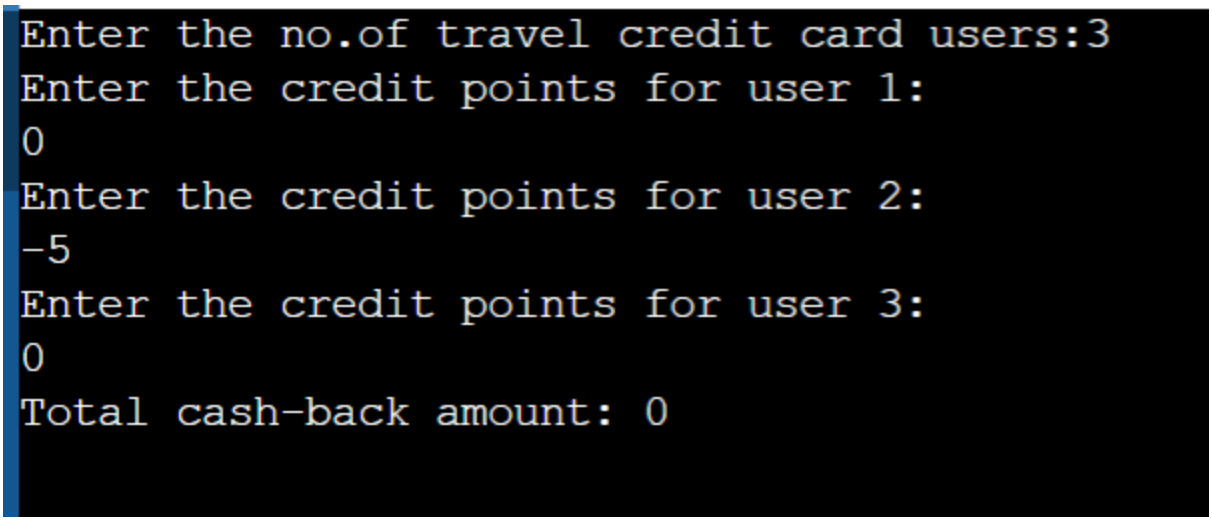
```
Enter the number of viewers:
5
2
3
4
8
9
The highest rating is for 0-5
```

28.Cash Back Offer

```
def calculate_amount(credit_points):
    sum=0
    for i in credit_points:
        if i>=50:
            sum+=i*5
        elif i>=30 and i<=50:
            sum+=i*2
        elif i<30 and i>=0:
            sum+=i*1
        else:
            sum+=0
    print("Total cash-back amount:",sum)

n=int(input("Enter the no.of travel credit card users:"))
credit_points=[]
for i in range(n):
    c=int(input(f"Enter the credit points for user {i+1}:\n"))
    credit_points.append(c)
calculate_amount(credit_points)
```

```
Enter the no.of travel credit card users:5
Enter the credit points for user 1
10
Enter the credit points for user 2
20
Enter the credit points for user 3
30
Enter the credit points for user 4
40
Enter the credit points for user 5
50
Total cash-back amount: 420
```




27.Analyze comments

```
def analyze_comments(input_string, keywords):
    isp = input_string.split('\\n')
    print (isp)
    comments = []
    for i in isp:
        username, comment = i.split(":")
        comments.append(comment)
    for i in comments:
        for keyword in keywords:
            if keyword in i:
                print(i)

input_string = input()
keywords=list(map(str,input().split(",")))
analyze_comments(input_string,keywords)
```

Edit this code

26.Dans scorecard

```
def calculate_score(score_values):
    c=0
    avg=sum(score_values)/score_value
    f=avg/2
    for i in score_values:
        if (i>=avg or i>f):
            c+=1
    print("The Score values that are equal to or above 50% of the average score:",c)
score_value=int(input("Enter the size of the score card"))
score_values=[]
print("Enter the score values:")
for _ in range(score_value):
    number = input()
    if "." in number:
        number = float(number)
        score_values.append(number)
    else:
        number = int(number)
        score_values.append(number)
calculate_score(score_values)
```

```
Enter the size of the score card3
Enter the score values:
0.2
0.5
1.5
The Score values that are equal to or above 50% of the average score: 2
```

25 Scholarships

Print output (drag lower right corner to resize)

```
117,112,113,114,115,116,111
113,114,115
Students without scholarships,117,112,116,111
```

```
def check_scholarships(string1,string2):
    a=[]
    if(len(string1)<len(string2)):
        print("Invalid data")
    else:
        if string1==string2:
            print("All students have scholarships")
        else:
            for i in string1:
                if i not in string2:
                    a.append(i)
            print("Students without scholarships",*a,sep=",")


string1=list(map(str,input().split(",")))
string2=list(map(str,input().split(",")))
check_scholarships(string1,string2)
```

Print output (drag lower right corner to resize)

```
117,112,113,114,115,116,111
117,112,113,114,115,116,111
All students have scholarships
```

```
117,112,113
117,112,113,114,115,116,111
Invalid data
```

24 Registration Number

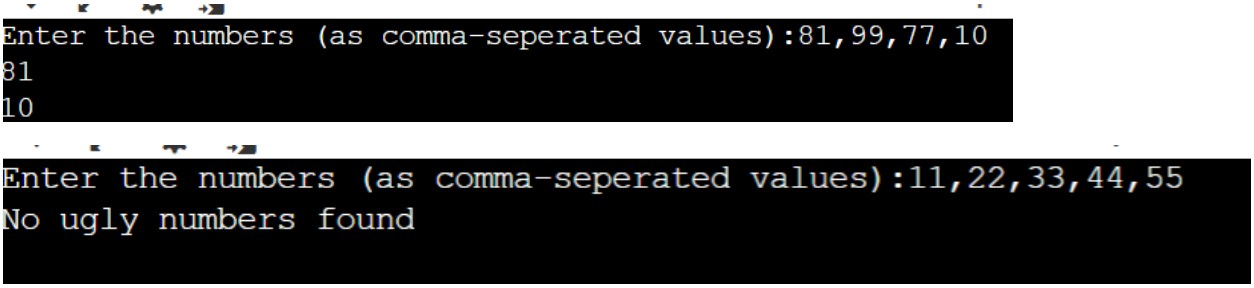
```
def filter_regno(reg_no):
    a='7119'
    x=[i for i in reg_no if not i.startswith(a)]
    print("Register numbers of students from other universities:",x)
n=int(input("Enter the no.of studnets registered for the webinar:"))
reg_no=[]
for i in range(n):
    k=input()
    reg_no.append(k)
filter_regno(reg_no)
```

```
Enter the no.of studnets registered for the webinar:5
7110987272
711974923874
89862391
71198765456
7899764357
Register numbers of students from other universities: ['7110987272', '89862391', '7899764357']
```

```
Enter the no.of studnets registered for the webinar:2
71198098
7119e45
Register numbers of students from other universities: []
```

23 Ugly Number

```
def is_ugly(number_list):
    ugly=[]
    for i in number_list:
        temp=i
        while i%2==0:
            i=i/2
        while i%3==0:
            i=i/3
        while i%5==0:
            i=i/5
        if i==1:
            ugly.append(temp)
    if len(ugly)==0:
        print("No ugly numbers found")
    else:
        for i in ugly:
            print(i)
number_list=list(map(int,input("Enter the numbers (as comma-seperated values):").split(",")))
is_ugly(number_list)
```




```
Enter the numbers (as comma-seperated values):81,99,77,10
81
10
```

```
Enter the numbers (as comma-seperated values):11,22,33,44,55
No ugly numbers found
```

22 Parking Details

21 Predict disesase probability

```python
def compute_risk_score(patient_data):
    sum=0
    if int(patient_data[0])>60:
        sum+=10
    else:
        sum+=5
    if patient_data[1]=='M':
        sum+=5
    else:
        sum+=3
    if int(patient_data[2])>120:
        sum+=10
    else:
        sum+=5
    if int(patient_data[3])>200:
        sum+=15
    else:
        sum+=10
    predict_probability(sum)

def predict_probability(risk_score):
    p=1-(1/(1+risk_score))
    print(f"The probability of the patient developing the disease is {p:.2f}")

patient_data=input("Enter the patient data \n").split(":")
compute_risk_score(patient_data)
```

```
Enter the patient data
62:M:125:210
The probability of the patient developing the disease is 0.98
```

```
Enter the patient data
44:F:100:180
The probability of the patient developing the disease is 0.96
```

20 Immunization Record

```
def create_record(children_records):
    cr=[]
    for i in children_records:
        a={}
        name,gender,weeks,contact=i.split(':')
        a['Name']=name
        a['Gender']=gender
        a['weeks']=weeks
        a['Contact']=contact
        cr.append(a)
    return cr
def display_record(valid_records,weeks):
    c=0
    c1=1
    for i in valid_records:
        print("Records",i)
        if int(i['weeks'])>=weeks:
            for key,value in i.items():
                print(f"{key}:{value}")
                c+=1
                c1+=1
    print(f"There are {i} children under {weeks} who have booked for the v
n=int(input("Enter name,gender,weeks, and contact as colon-seperated value
b=[]
for i in range(n):
    k=input()
    b.append(k)
o1=create_record(b)
w=int(input("To display the records based on "))
o2=display_record(o1,w)
```

19 Task Manager

```
def add_task(task,todo_list):
    todo_list.append(task)
    return todo_list
def mark_task(index,todo_list):
    if len(todo_list)>index:
        del todo_list[index]
        print(*todo_list,sep="\n")
    else:
        print("Invalid input")
todo_list=[]
while 1:
    n=int(input("Enter a command(1 to add a task,2 to mark a task,3 to qu
    if n==1:
        i=input("Enter a task to add:")
        add_task(i,todo_list)
    elif n==2:
        g=int(input("Enter the index of the task to mark as complete:"))
        ou=mark_task(g,todo_list)
    elif n==3:
        break;
    else:
        print("Invalid command")
```

18 Game Event

```
from collections import *
def find_each_round_winner(team1,team2):
    k=[]
    for i in range(len(team1)):
        if team1[i]>team2[i]:
            k.append('Team1')
        elif team1[i]<team2[i]:
            k.append("Team2")
        elif team1[i]==team2[i]:
            k.append('Equal')
        else:
            print("None")
    return k
def count_winners(winner_list):
    c=Counter(winner_list)
    return c
n=int(input("Enter the no.of rounds"))
if(n<0):
    print("Invalid Rounds")
else:
    print("Enter the team 1 points:\n")
    team1=[]
    for i in range(n):
        k=int(input())
        team1.append(k)
    team2=[]
    print("Enter the team 2 points:\n")
    for i in range(n):
        k=int(input())
        team2.append(k)
    a=find_each_round_winner(team1,team2)
    print(a)
    b=count_winners(a)
    print("Team1:",b['Team1'])
    print("Team2:",b['Team2'])
    print("Equal:",b['Equal'])
```

17 Cricket Academy

```
def create_player(player_id,player_name,matches_played,runs_scored):
    a={}
    a['Id']=player_id
    a['Name']=player_name
    a['Matches Played']=matches_played
    a['Runs Scored']=runs_scored
    return a
def display_players(player_details):
    c=0
    for i in player_details:
        v=1
        print(f"Player {v}")
        for key,value in i.items():
            if i['Runs Scored']>100:
                print(f"{key}:{value}")
                c+=1
                v+=1
    if c==0:
        print("No details Found")
f=[]
while 1:
    print('''1.create_player
    2.Display Player details
    3.Exit''')
    n=int(input("Enter the option:"))
    if n==1:
        player_id=input("player id")
        player_name=input("player name")
        matches_played=input("matches played")
        runs_scored=int(input("Runs Scored"))
        f.append(create_player(player_id,player_name,matches_played,runs_s
    elif n==2:
        display_players(f)
    else:
        print("Thank You")
        break
```

16 Deducing Blood group

```
def create_list(factors):
    factors=factors.split(",")
    return factors
def deduce_blood_group(blood_details):
    if a[0]==a[2]:
        return False
    elif a[1]==a[3]:
        return  False
    else:
        if (a[0]==a[3]) and a[4]=='+':
            return 'A+'
        elif (a[1]==a[2]) and a[4]=='+':
            return 'B+'
        elif (a[0]==a[1]) and a[4]=='+':
            return 'AB+'
        elif (a[2]==a[3]) and a[4]=='+':
            return 'O+'
        elif (a[0]==a[3]) and a[4]=='-':
            return 'A-'
        elif (a[1]==a[2]) and a[4]=='-':
            return 'B-'
        elif (a[0]==a[1]) and a[4]=='-':
            return 'AB-'
        elif (a[2]==a[3]) and a[4]=='-':
            return 'O-'
        else:
            print("Invalid options")
b=input("Enter y/n for A antigens,y/n for B antigens,y/n for anti-A antibo
a=create_list(b)
fo=deduce_blood_group(a)
if fo:
    print("Deduced blood group:",fo)
else:
    print("Incorrect combination of antigens/antibodies entry")
```

```
Enter y/n for A antigens,y/n for B antigens,y/n
Deduced blood group: AB-
```

```
Enter y/n for A antigens,y/n for B antigens,y/n
Incorrect combination of antigens/antibodies entr
```

15 Bike Race

```
def calculate_time(race_details):
    aa=[]
    for i in race_details:
        bid,name,speed=i.split(":")
        a={}
        tt=round((200/int(speed)),1)
        a['Id']=bid
        a['Name']=name
        a['Time']=tt
        aa.append(a)
    return aa
def find_qualifiers(race_details,time):
    o=[]
    for i in race_details:
        for key,value in i.items():
            if key=='Time':
                if value>=time:
                    o.append(i['Name'])
    return o

n=int(input("Enter the no.of race participants"))
race_details=[]
for i in range(n):
    k=input()
    race_details.append(k)
a=calculate_time(race_details)
b=float(input("Enter the time to qulaify for the next level"))
c=find_qualifiers(a,b)
if len(c)>0:
    print(*c,sep="\n")
else:
    print("No one is qualified")
```

```
Enter the no.of race participants3
BK12:Keane:80
BK03:Maxi:85
BK23:Smith:60
Enter the time to qulaify for the next level2.5
Keane
Smith
```

14 Room Rent

```
def calculate_days(from_date,to_date):
    fd,fm=from_date.split('/')
    td,tm=to_date.split('/')
    fd=int(fd)
    td=int(td)
    fm=int(fm)
    tm=int(tm)
    if(fm==tm):
        nd=td-fd
    else:
        if (fm!=tm):
            nm=tm-fm
            nd1=30-fd
            nd=((nm-1)*30)+nd1+td
    return nd
def calculate_total_amount(customer_name,room_type,no_of_days):
    d={}
    if (room_type=='Single') and (no_of_days<=3):
        tm=3300-(3300*0.1)
    elif (room_type=='Single') and (no_of_days>3):
        tm=3300-(3300*0.15)
    elif (room_type=='Double') and (no_of_days<=3):
        tm=4000-(4000*0.1)
    elif (room_type=='Double') and (no_of_days>3):
        tm=4000-(4000*0.17)
    elif (room_type=='Triple') and (no_of_days<=3):
        tm=4500-(4500*0.1)
    elif (room_type=='Triple') and (no_of_days>3):
        tm=4500-(4500*0.2)
    else:
        print("Invalid room type")
    d['Customer name']=customer_name
    d['No.of.days']=no_of_days
    d['Total amount']=tm
    return d


a=input("Enter details")
roomno,name,rtype,fdate,tdate=a.split(":")
no_of_days=calculate_days(fdate,tdate)
f2=calculate_total_amount(name,rtype,no_of_days)
for key,value in f2.items():
    print(f"{key}:{value}")
```

```
Enter detailsARO123:Smith:Double:12/05:13/05
Customer name:Smith
No.of.days:1
Total amount:3600.0
```

13 Filter Participants

```
def calculate_score(participants_list):
    c={}
    for i in participants_list:
        name,s1,s2,s3=i.split(":")
        avg=(int(s1)+int(s2)+int(s3))//3
        c[name]=avg
    return c
def Filter_participants(participants_dictionary,pass_score):
    ps=[]
    for key,value in participants_dictionary.items():
        if value>=pass_score:
            ps.append(key)
    return ps
n=int(input("Enter the no.of participants:"))
participants_list=[]
print("Enter the details")
for i in range(n):
    k=input()
    participants_list.append(k)
f1=calculate_score(participants_list)
pass_score=int(input("Enter the pass score to select next level:"))
ps=Filter_participants(f1,pass_score)
if len(ps)<=0:
    print("No one selected")
else:
    print(*ps,sep="\n")
```

12 Trainer Ratings

```
def create_ratings(input_string):
    input_string=input_string.split(",")
    a={}
    for i in input_string:
        tid,value=i.split(':')
        a[tid]=float(value)
    return a
def count_ratings(rating_dict):
    f=[]
    s=[]
    for key,value in rating_dict.items():
        if value>0 and value <=5:
            f.append(key)
        else:
            s.append(key)
    return f,s


input_string=input("Enter the ratings (as comma-seperated values):")
B=create_ratings(input_string)
f,s=count_ratings(B)
if len(f)==0:
    print("The list of trainers with ratings between 0-5 Nil")
else:
    print("The list of trainers with ratings between 0-5",f)
if len(s)==0:
    print("The list of trainers with ratings between 6 an above Nil")
else:
    print("The list of trainers with ratings between 6 and above",s)
```

```
Enter the ratings (as comma-seperated values):WW12:5,TT11:9,WQ34:7,YU60:3,BN01:7,VV55:8,PL23:6
The list of trainers with ratings between 0-5 ['WW12', 'YU60']
The list of trainers with ratings between 6 and above ['TT11', 'WQ34', 'BN01', 'VV55', 'PL23']
```

```
Enter the ratings (as comma-seperated values):WW12:5,YU60:3
The list of trainers with ratings between 0-5 ['WW12', 'YU60']
The list of trainers with ratings between 6 an above Nil
```

11 Key Generation

```
def generate_secret_key(name):
    name=name.lower()
    sum=0
    if name.isalpha() and (len(name)>2 and len(name)<=10):
        for i in name:
            sum+=ord(i)
        avg=sum//(len(name))
        print(chr(avg))
    else:
        print("Invalid Input")
name=input("Enter the name:")
generate_secret_key(name)
```

```
Enter the name:Mathew
k
```

```
Enter the name:Ab123
Invalid Input
```

10 Replace word

```
def replace_word(sentence,word):
    sentence=sentence.split(" ")
    if word not in sentence:
        print("The word is no there in sentence")
    else:
        a=['*'*len(word)  if i.lower()==word.lower() else i for i in sentence]
        print(*a,end=" ")

sentence=input("Enter the sentence:\n")
word=input("Enter the word")
replace_word(sentence,word)
```

9 Product code

```
and int(m)<=12) and len(y)==4):
```

```
def generate_code(product_details):
    pname,des,m,y=product_details.split(":")
    a=[]
    if (len(pname)>3 and len(des)>3 and (int(m)>=1 and int(m)<=12) and le
        if len(pname)%2==0:
            a.append(pname[-3::].upper())
        elif len(pname)%2==1:
            a.append(pname[0:3].upper())
        else:
            pass
        k=des[0].upper()
        f=des[-1].upper()
        v=k+f
        a.append(v)
        mo=m[0::]
        ye=y[-2::]
        moye=mo+ye
        a.append(moye)
    else:
        print("Invalid product details")
    return a


product_details=input("Enter the details:\n")
b=generate_code(product_details)

print(*b,sep="/")
```

8 Find Grade

```
def find_exam_result(correct,incorrect):
    if correct+incorrect>120 or correct+incorrect<120:
        print("Invalid number of questions")
        exit()
    else:
        correct=correct*2
        incorrect=incorrect*1
    sum=correct-incorrect
    per=(sum/120)*100
    if per>=75:
        print("You have received A grade")
    elif per>=60 and per<75:
        print("You have received B grade")
    elif per>=50 and per<60:
        print("You have received c grade")
    else:
        print("Sorry!You have failed")
correct=int(input("Enter the count of correct answers:"))
incorrect=int(input("Enter the count of incorrect answers:"))
find_exam_result(correct,incorrect)
```

```
Enter the count of correct answers:90
Enter the count of incorrect answers:30
You have received A grade
```

7 Toll Checkl

```
def check_number(vehicle_number):
    if len(vehicle_number)<10 or len(vehicle_number)>10:
        print("Invalid vehicle number")
        exit()
    a=[]
    for i in vehicle_number:
        if i.isdigit():
            a.append(int(i))
    if sum(a[-4:])%2==0:
        disc=a[-4]+a[-2]
    elif sum(a[-4:])%2==1:
        disc=a[-3]+a[-1]
    else:
        print("Invalid")
    print("Your discount percentage is",disc)

vehicle_number=input("Enter Vehicle Number:")
check_number(vehicle_number)
```

```
Enter Vehicle Number:TN43AD23
Invalid vehicle number
```

6 Lucky Number

```
def find_lucky_number(dob):
    a=dob.split("/")
    if '/' not in dob:
        print("Invalid format")
    if int(a[0])>=1 and int(a[0])<=31:
        if int(a[1])>=1 and int(a[1])<=12:
            if int(a[2])<=2023:
                sum=int(a[0])+int(a[1])+int(a[2])
                sum1=0
                while(sum>0):
                    r=sum%10
                    sum1+=r
                    sum=sum//10
                print(f"The lucky number is {sum1}")
            else:
                print("Invalid format")
        else:
            print("Invalid format")
    else:
        print("Invalid format")

dob=input("Enter the date of birth:\n")
find_lucky_number(dob)
```

```
Enter the date of birth:
11-11-2001
```

```
Enter the date of birth:
24/04/1999
The lucky number is 11
```

5 Flat Discount

```
def calculate_discount(input_string):
    a=input_string.split(":")
    n=int(a[0])
    sum=0
    while(n>0):
        r=n%10
        sum=sum+r
        n=n//10
    if n%2==0:
        if a[1]=='2BHK':
            disc=(3700000/100)*5
        else:
            disc=(4900000/100)*7
    else:
        if a[1]=='2BHK':
            disc=(3900000/100)*4
        else:
            disc=(5100000/100)*8
    print(disc)
input_string=input("Enter the details")
calculate_discount(input_string)
```

```
Enter the details435:3BHK
343000.0
```

4.Blood Pressure Status

```
def generate_status(BP_level):
    a=BP_level.split('/')
    a=list(map(int,a))
    if a[0]<90 and a[1]<60:
        print("Low BP")
    elif (a[0]>=90 and a[0]<=120) and (a[1]>=60 and a[1]<=80):
        print("Normal")
    elif (a[0]>=121 and a[0]<=140) and (a[1]>=81 and a[1]<=90):
        print("Pre-High BP")
    elif (a[0]>=141 and a[0]<=190) and (a[1]>=91 and a[1]<=100):
        print("High BP")
    elif a[0]>190 and a[1]>100:
        print("Hyper Tension")
    else:
        print("Invalid Input")

BP_level=input("Enter the BP level:")
generate_status(BP_level)
```

3.Sentence Palindrome

```
def is_palindrome(sentence):
    a=sentence.split()
    rev=' '.join(a[::-1])
    if rev==sentence:
        return True
    return False
sentence=input("Enter the sentence:")

if is_palindrome(sentence):
    print(f"{sentence} is palindrome")
else:
    print(f"{sentence} is not a palindrome")
```

```
string = ''.join([char for char in string if
char.isalnum()])
```

```
Enter the sentence:lirill,mom lirill
lirill,mom lirill is palindrome
```

```
Enter the sentence:sunrises in the east
sunrises in the east is not a palindrome
```

2.Find Relationship

```
def find_relationship(name1,name2):
    name1=name1.replace(" ","")
    name2=name2.replace(" ","")
    sum=len(name1)+len(name2)
    val=sum%6
    if val==0:
        print("Soulmates")
    elif val==1:
        print("Colleagues")
    elif val==2:
        print("Friends")
    elif val==3:
        print("Good friends")
    elif val==4:
        print("Best friends")
    elif val==5:
        print("Close friends")
    else:
        print('')

name1=input("Enter the name 1:")
name2=input("Enter the name 2:")
find_relationship(name1,name2)
```

```
Enter the name 1:Glenn
Enter the name 2:Kim
Friends
```

```
Enter the name 1:Lilly
Enter the name 2:Lenny
Best friends
```

```
Enter the name 1:Glenn Martin
Enter the name 2:Liv Morgan
Friends
```

1.Validating Aliteration

```
def is_alliterative(sentence):
    a=[]
    c=['a','e','i','o','u']
    sentence=sentence.split(" ")
    if len(sentence)<2:
        print("The sentence is not alliterative")
    for i in sentence:
        a.append(i[0])
    if c in a:
        print("The sentence is not alliterative")
    if len(set(a))>1:
        print("The sentence is not alliterative")
    else:
        print("The sentence is alliterative")



sentence=input("Enter the sentence to be validated fror alliteration:")
is_alliterative(sentence)
```

```
Enter the sentence to be validated fror alliteration:she sells sea shells
The sentence is alliterative
```

```
Enter the sentence to be validated fror alliteration:Ann sells sea shells
The sentence is not alliterative
```