I’m not sure if a report is written like this, THANKS FOR YOUR READING!

# 1. Flowchart

```
    while a > b :
        if b > c :
            print("No value to print.")
            break
        else :
            x = c
            y = b
            z = a
            result = x+y-10*z
            print("The value is "+ str(result) + ".")
            break
    while a <= b :
        if b > c :
            x = a
            y = b
            z = c
        if (b <= c)&(a > c) :
            x = c
            y = a
            z = b
        else :
            x = a
            y = c
            z = b
        result = x+y-10*z
        print("The value is "+ str(result) + ".")
        break
```

Print_values("5","15","10")

#The value is -135.

# 2. Continuous celing function

#resources:

#1.https://zhuanlan.zhihu.com/p/626462316

#2.https://www.delftstack.com/zh/howto/python/sort-list-by-another-list-python/

#define the list filled with positive integers in a given range

```
N = int(input("How many numbers do you want in the list: "))
A = int(input("How large do you want the number can be: "))
#example:
# N = 10
# A = 15
```

```
my_list = []
for i in range(N):
    my_list.append(random.randint(1,A))
print("The list is " + str(my_list))

value_x = [0] * A
value_list = []
cp_list = sorted(my_list)
for x in range(A+1):
    if x == 1:
        value_x[x-1] = 1
        continue
    elif (x%3 != 0) :
        value_x[0] = 1
        y = int(ceil(x/3))
        value_x[x-1] = value_x[y-1] + x*2
    else:
        y = int(x/3)
        value_x[x-1] = value_x[y-1] + x*2

for x1 in my_list:
    value_list.append(value_x[x1-1])
print("The value of continuous ceiling function is: " + str(value_list))


# 3. Dice rolling
#3.1
#resources:
#sorry for citing the website too much
#https://tutorialspoint.org.cn/program-to-count-number-of-ways-we-can-throw-n-dices-in-pytho
n
def Find_number_of_ways(x):
    if x < 10 or x > 60:
        return 0
    # dp[i][j]
    n = 10
    dp = [[0] * (x + 1) for _ in range(11)]
    for m in range(1, 7):
        if m <= x:
            dp[1][m] = 1
    for i in range(2,11):
        for j in range(i,min(i*6, x)+1):
            for face in range(1,7):
                if j - face >= 1:
                    dp[i][j] += dp[i-1][j - face]

    return dp[n][x]
solution = ()
print(solution)
For example: Find_number_of_ways(12), the ways are 55.

#3.2
Number_of_ways = []
for x in range(10,61):
    Number_of_ways.append(Find_number_of_ways(x))
#x is starting from 10
print(str(Number_of_ways.index(max(Number_of_ways))+10) + " yields the maximum of.")
#plt.plot(Number_of_ways)



# 4. Dynamic programming
#4.1
def Random_integer(N):
    return [random.randint(0, 10) for _ in range(N)]

#Random_integer(10)

#4.2
#resource: https://blog.csdn.net/weixin_43509127/article/details/104394075
def Sum_averages(arra):
    subsets = []
    arra1 = set[arra]
    for k in range(N):
        k = 2
        subset = list(itertools.combinations(arra1, k))
        ave = np.mean(subset)
        subsets.append(subset)
    print(subsets)
#4.3


# 5. Path counting
#5.1
N = int(input("Input the number of rows: "))
M = int(input("Input the number of coloums: "))

a = np.random.randint(2, size = N*M).reshape(N,M)
```

```
a[0,0]=a[N-1,M-1] = 1
print(a)

#5.2
#source:https://zhuanlan.zhihu.com/p/43358393
def Count_path(a):
    N = len(a)
    M = len(a[0])
    count = [[0] * M for _ in range(N)]
    if a[0][0] == 1:
        count[0][0] = 1
    for i in range(N):
        for j in range(M):
            if a[i][j] == 0:
                continue
            if (i != 0) or (j != 0):
                top = count[i-1][j] if i > 0 else 0
                left = count[i][j-1] if j > 0 else 0
                count[i][j] = top + left
    return count[N-1][M-1]
Count_path(a)

#5.3
N = 10
M = 8
path_count = []
for x in range(1000):
    path_count.append(Count_path(matrix(N,M)))
sum = np.sum(path_count)
ave = sum/1000

print("The average count of paths is : "+ str(ave))
```