week 1

```python
#1
print('1')
print('11')
print('111')
print('1111')
print('11111')
```

```python
#2
print('   1')
print('  1 1')
print(' 1 2 1')
print('1 3 3 1')
```

```python
#3
a = input()
c = str(len(a))
d = a+c
print(d)
```

```python
#4
a = int(input())
c = (2**100)%a
print(c)
```

```python
#5
c = input()
print(c[0:2])
```

week 2

```python
a = int(input())
b = int(input())
c = int(input())
if (c*c==a*a+b*b or a*a==c*c+b*b or b*b==a*a+c*c) :
    print('YES',end='')
else:
    print('NO',end='')
```

```python
T = int(input())
if T>=0 and T<24:
    if T>=0 and T<=5:
        print("NIGHT",end="")
    if T>=6 and T<=11:
        print("MORNING",end="")
    if T>=12 and T<=17:
        print("AFTERNOON",end="")
    if T>=18 and T<=23:
        print("EVENING",end="")
else:
    print("INVALID", end="")
```

```python
a = int(input())
b = int(input())
c = int(input())
d = int(input())
e = int(input())
if (a+b)%2==0:
    if (b+c)%2==0:
        if (c+d)%2==0:
            if (d+e)%2==0:
                if (a+e)%2==0:
                    print('YES',end='')
                else:
                    print('NO',end='')
            else:
                print('NO',end='')
        else:
            print('NO',end='')
    else:
        print('NO',end='')
else:
    print('NO',end='')
```

```python
s = input()
if 'a' in s or 'A' in s:
    print('a',end='')
if 'e' in s or 'E' in s:
    print('e',end='')
if 'i' in s or 'I' in s:
    print('i',end='')
if 'o' in s or 'O' in s:
    print('o',end='')
if 'u' in s or 'U' in s:
    print('u',end='')
```

```python
P1 = input()
D1 = input()
P2 = input()
D2 = input()
Y1 = int(D1[6:11])
Y2 = int(D2[6:11])
M1 = int(D1[3:5])
M2 = int(D2[3:5])
d1 = int(D1[0:2])
d2 = int(D2[0:2])
if D1 != D2:
    if Y1 != Y2:
        if Y1 > Y2:
            print(P1,end='')
        else:
            print(P2,end='')
    elif M1 != M2:
        if M1 > M2:
            print(P1,end='')
        else:
            print(P2,end='')
    else:
        if d1 > d2:
            print(P1,end='')
        else:
            print(P2,end='')
else:
    if P1 < P2:
        print(P1,end='')
    else:
        print(P2,end='')
```

```python
P = input()
notAlpha = "1234567890!@#$%^&_*./\='\" "
S = "/\='\" "
C = True
if len(P)>=8 and len(P)<=32:
    if P[0] not in notAlpha:
        for a in P:
            if a in S:
                C = False
        print(C,end='')
    else:
        print(False,end='')
else:
    print(False,end='')
```

week 3

```python
#1
num=int(input())
sum=0
for i in range(1,num+1):
    for j in range(1,1+i):
        sum=sum+j
print(sum)
```

```python
#2
n = int(input())
d = []
for i in range(2, n+1):
    flag = True
    for j in range(2, i):
        if (i % j == 0):
            flag = False
            break
    if flag:
        d = d + [i]
for a in d:
    if n%int(a) == 0:
        print(a)
```

```python
#3
text = input().lower()
length = len(text)
output_string = ""
for char in "abcdefghijklmnopqrstuvwxyz":
    for i in range(0, length):
        if char == text[i]:
            output_string += char
print(output_string)
```

```python
#4
x = input()
for i in range(2, len(x)):
    flag = True
    for j in range(2, i):
        if (i % j == 0):
            flag = False
            break
        if flag:
            print(x[i])
```

```python
#5
num = input()
startnum = '6789'
status = False
if num.isdigit():
        if num[0] in startnum:
                if len(num) == 10:
                        flag = True
                        for i in range(0, 6):
                                if 5 * num[i] in num:
                                        flag = False
                        if flag:
                                flag2 = False
                                for j in num:
                                        if (num.count(j) < 7): flag2 = True
                                if flag2:
                                 print('valid',end='')
                                 status = True
if not(status):
  print('invalid',end='')
```

week 4

```python
#1
S, l = 0, []
x = input()
while x != 'END':
    l.append(float(x))
    x = input()
if len(l) > 1:
    avg = sum(l) / len(l)
    for i in l:
        S += (i-avg)**2
    SD = (S / (len(l)-1))**0.5
    print(f'{SD:.2f}')
```

```python
#2
s = input()
current_max = 0
max = 0
flag = True
n = len(s)
for i in range(n):
  if s[i] == '(':
    current_max += 1
    if current_max > max:
        max = current_max
  elif s[i] == ')':
    if current_max > 0:
        current_max -= 1
    else:
        flag = False
        break
if not(flag) or current_max != 0:
  print('Not matched',end='')
else:
  print(max,end='')
```

```python
#3
l = []
n = input()
while n:
    l.append(int(n))
    n = input()
l.sort()
for i in range(len(l)):
    for j in range(len(l)):
        if l[i] + l[j] in l and i != j:
            print(l[i], l[j])
```

```python
#4
mat = []
flag = True
new_mat = []
while flag:
  row = input().split(' ')
  if row == [""]:
    break
  mat.append(row)
col = len(mat[0])
box = []
for i in range(col-1,-1,-1):
  for j in range(len(mat)):
     box.append(mat[j][i])
  new_mat.append(box)
  box = []
for a in new_mat:
  print(' '.join(a))
```

week 5

```python
#1
def perfect_number(num):
    l = []
    for i in range(1,num):
        if  num % i == 0:
            l.append(i)
    sum = 0
    for j in l:
        sum += j
    if sum == num:
        return True
    else:
        return False
```

```python
#2
def user_score(read_count,reply_count,new_post_count):
    sum = read_count + 3*reply_count + 5*new_post_count
    if sum > 50:
        return "Leader"
    else:
        return "Basic"
```

```python
#3
def check_leap_year(year):
    if (year % 400 == 0) or (year % 100 != 0 and year % 4 == 0):
        return True
    else:
        return False
```

```python
#4
def is_magic(mat):
    m = len(mat)
    d1sum, d2sum = 0, 0
    for i in range(m):
        d1sum += mat[i][i]
        d2sum += mat[i][m - i - 1]
    if not(d1sum == d2sum):
        return 'NO'
    for i in range(m):
        rsum, csum = 0, 0
        for j in range(m):
            rsum += mat[i][j]
            csum += mat[j][i]
        if not(rsum == csum == d1sum):
            return 'NO'
    return 'YES'
```

```python
#5
def process():
    first = board_min()
    board_erase(first)

    if board_isEmpty():
        return first

    second = board_min()
    board_erase(second)
    delta = first - second if first > second else second - first
    board_write(delta)

    return process()
```

week 6

```python
#1
def freqWords(words):
    char = ('.',',',':',';','!')
    List = []
    d = {}
    for k in words:
        for i in char:
            if i in k:
                k = k.replace(i,'')
        List.append(k.lower())


    for a in List:
        d[a]=0

    for a in List:
        d[a] += 1
    s = {}
    for b in d:
        count = d[b]
        if count not in s:
            s[count] = []
        s[count].append(b)
    sorted_dict = {}
    for i in set(s):
        sorted_dict[i] = sorted(s[i])
    return sorted_dict
```

```python
#2
def crowdedGroup(scores,subject,markLimit):
    List = []
    for i in scores:
            List.append([i['SeqNo'],i[subject]])
    P = []
    for i in List:
        P.append(i[1])
    g = {}
    for i in P:
        g[i] = []
        for j in P:
            if j <= i+int(markLimit) and j >= i:
                g[i].append(j)
    max = 0
    for a in P:
        if max < len(g[a]):
            max = len(g[a])
    E = []
    F = []
    for b in g:
        if len(g[b]) == max:
            for i in scores:
                if i[subject] in g[b]:
                    E.append(i['SeqNo'])
            F.append(E)
            E = []
    return sorted(F)
```

```python
#3
def topMentors(scores,subject):
    List = []  #taking Roll and Sub_Mark
    for i in scores:
            List.append([i['SeqNo'],i[subject]])
    P = [] #Sub_MArk
    for i in List:
        P.append(i[1])
    g = {} #Dict with Mark as key
    for i in P:
        g[i] = []
        for j in P:
            if j < i and (i-j >= 10) and (i-j <= 20):
                g[i].append(j)
    max = 0
    for a in P:
        if max < len(g[a]):
            max = len(g[a])
        E = []
        F = {}
        for b in g:
            if len(g[b]) == max:
                for i in scores:
                    if i[subject] == b:
                        k = i['SeqNo']
                        F[i['SeqNo']] = []
                for i in scores:
                    if i[subject] in g[b]:
                        E.append(i['SeqNo'])
                F[k] = E
                E = []
    return F
```

week 7

```python
#1
words = ('zero','one','two','three','four','five','six',
         'seven','eight','nine')
num = input()
List = []
for i in num:
    print(words[int(i)])
    List.append(words[int(i)])
string = ''
for j in List:
    if string == '':
        string += j
    else:
        j = j.capitalize()
        string += j
print(string)
```

```python
#2
TEST_CASE = input()
n = int(input())
List = input().split(',')
students = {}
for i in range(n):
    DATA = input().split(',')
    id = int(DATA[0])
    students[id] = {}
    for j in range(1,len(List)):
        students[id][List[j]] = int(DATA[j])
```

```python
#3
def merge(D1,D2,priority):
    if priority == 'first':
        D2.update(D1)
        return D2
    if priority == 'second':
        D1.update(D2)
        return D1
```

```python
#4
m  = int(input())
n  = int(input())
mat = []
for i in range(m):
    a = input().split(' ')
    mat.append(a)
for j in mat[1:-1]:
    for p in j[1:-1]:
        j[j.index(p)] = '0'
for k in range(m):
    print(*mat[k])
```

```python
#5
def collatz_repeat(n):
    if n==1:
        return 0
    else:
        if n%2==0:
            return 1 + collatz_repeat(n//2)
        else:
            return 1 + collatz_repeat(3*n+1)
```

week 8

```python
#1
def reverse(input_list):
    if len(input_list) == 0:
        return input_list
    last = input_list[-1]
    input_list.remove(input_list[-1])
    return [last] + reverse(input_list)
```

```python
#2
def max_element(input_list):
    if len(input_list) == 1 :
        return input_list[0]
    else:
        return max(input_list[0],
                   max_element(input_list[1:len(input_list)]))
```

```python
#3
def simple_sort(item_list):
    for i in range(len(item_list)):
        for j in range(i,len(item_list)):
            if item_list[i] > item_list[j]:
                item_list[i],item_list[j] = item_list[j],item_list[i]
    return item_list

def simple_search(item_list,item):
    L,n = item_list,item
    if len(L) == 1 and L[0] != n:
        return False
    if len(L) == 1 and L[0] == n:
        return True
    if len(L) == 2:
        for i in L:
            if i == n:
                decision = True
            else:
                decision = False
        return decision
    if len(L) > 2:
        if L[(len(L)-1)//2] == n:
            return True
        if L[(len(L)-1)//2] > n:
            L = L[0:((len(L)-1)//2)]
            return simple_search(L,n)
        if L[(len(L)-1)//2] < n:
            L = L[((len(L)-1)//2)+1:len(L)]
            return simple_search(L,n)
```

```python
#4
def add_movie_to_boxoffice(movies_db,new_movies):
    movies_db[new_movies[0]] = new_movies[1:3]
    return movies_db

def total_collection(movies_db):
    L = list(movies_db.values())
    K = []
    for i in L:
        K.append(i[0])
    return sum(K)

def average_collection(movies_db):
    L = list(movies_db.values())
    K = []
    for i in L:
        K.append(i[0])
    mean = sum(K)/len(K)
    return round(mean,2)

def num_of_movies_above_average_movies(movies_db):
    L = list(movies_db.values())
    K = []
    for i in L:
        K.append(i[0])
    mean = sum(K)/len(K)
    M = []
    for j in movies_db.keys():
        if movies_db[j][0] > mean:
            M.append(j)
    return len(M)

def highest_grossing_movie_year(movies_db):
    L = list(movies_db.values())
    K = []
    for i in L:
        K.append(i[0])
    MAX = max(K)
    for j in movies_db.keys():
        if movies_db[j][0] == MAX:
            return movies_db[j][1]
```

```python
#5
def trending(subject_topics):

 #Removing duplicates items-----------------------

 for i in subject_topics:
    j = subject_topics.index(i)
    k = set(i)
    subject_topics.remove(i)
    subject_topics.insert(j,list(k))

 #Initialising
 common_topics_list= []
 S = {}

 #Collating all items common_topics_list------------
 for a in subject_topics:
     for b in a:
         if b not in common_topics_list:
            common_topics_list.append(b)

 for c in common_topics_list:
    count = 0
    for a in subject_topics:
        for b in a:
            if c == b:
                count += 1
    if count not in S.keys():
       S[count] = [c]
    else:
        S[count].append(c)
 top_trend = len(S[max(S.keys())])
 least_trend = len(S[min(S.keys())])
 return top_trend,least_trend
```

week 9

```python
#1
def solution():
    '''
    0. Read the file.
    1. Accept input from the user as specified in the question.
    2. The input will be on three lines.
    3. For each of the three questions given in the problem statement,
       print your answer.
    '''
    f = open('WorldPopulation.csv','r')
    head = f.readline().strip().split(',')
    a = int(input())
    b = int(input())
    c = input()
    List = []
    lines = f.readlines()
    D = {}
    for line in lines[0:]:
        List.append(line.strip().split(','))
    for i in List:
        D[int(i[0])] = i[1:len(i)]
    print(int(D[a][0]))
    List_2 = []
    for key in D:
        if int(D[key][0]) > b:
            List_2.append(key)
    print(int(min(List_2)))
    List_3 = []
    for b in List:
        List_3.append(float(b[head.index(c)]))
    print(max(List_3))
    f.close()
```

```python
#2
def highest_grossing(yearFrom, yearUpto, genre):
    '''
    Arguments:
        yearFrom: int
        yearUpto: int
        genre: string
    Returns:
        movie_name: string
    '''
    f = open('IMDB_reviews.csv','r')
    List = []
    lines = f.readlines()
    for line in lines[1:]:
            List.append(line.strip().split(','))
    List_1 = []
    for i in List:
        if int(i[2]) >= yearFrom and int(i[2]) <= yearUpto:
            List_1.append(i)
            #print(i[2]----------------------)
    #print(len(List_1)-----------------------)
    List_2 = []
    for i in List_1:
        #print(i[4]--------------------------)
        if genre in i[4] :
            List_2.append(i)
            #print(i[0],i[1],i[4]-----------)
    #print(List_2---------------------------)
    List_3 = []
    for i in List_2:
        if i[9] != '':
            List_3.append(int(i[9]))
    #print(List_3,---------checkpoints------)

    for i in List_2:
        if i[9] == str(max(List_3)):
            f.close()
            return i[1]
```

```python
#3
def solution():
    '''
    1. Process version_1.txt and version_2.txt
    2. Print the number lines in version_2.txt that are not in version_1.txt
    '''
    f = open('version_1.txt','r')
    g = open('version_2.txt','r')
    Lines_1 = f.readlines()
    Lines_2 = g.readlines()
    L1 = []
    for line in Lines_1:
        L1.append(line.strip())
    L2 = []
    for line in Lines_2:
        L2.append(line.strip())
    print(len(set(L2)-set(L1)))
    f.close()
    g.close()
```

week 10

```python
#1
class Point :
    def __init__(self,x = 0,y = 0):
        self.x = x
        self.y = y
    def move(self,dx = 0,dy = 0):
        self.x += dx
        self.y += dy
    def value(self):
        return self.x,self.y
    def duplicate(self):
        return Point(self.x,self.y)
```

```python
#2
import math
class Point :
    def __init__(self,x = 0,y = 0):
        self.x = x
        self.y = y
    def move(self,dx = 0,dy = 0):
        self.x += dx
        self.y += dy
    def value(self):
        return self.x,self.y
    def duplicate(self):
        return Point(self.x,self.y)
class Line:
    def __init__(self,A,B):
        self.A, self.B = Point.value(A),Point.value(B)
    def length(self):
        return math.hypot(self.A[0]-self.B[0],self.A[1]-self.B[1])
    def slope(self):
        if self.A[0]-self.B[0] != 0:
            return (self.A[1]-self.B[1])/(self.A[0]-self.B[0])
        return math.inf
```

```python
#3
class TimeConverter:
    def __init__(self,value):
        self.sec = value
    def Second_to_Minutes(self):
        m  = self.sec//60
        mx = self.sec%60
        s = mx
        return f'{m} min {s} sec'
    def Second_to_Hours(self):
        h = self.sec//(60*60)
        hx = self.sec%(60*60)
        m = (hx)//60
        mx = hx%60
        s = mx
        return f'{h} hr {m} min {s} sec'
    def Second_to_Days(self):
        d = self.sec//(60*60*24)
        dx = self.sec%(60*60*24)
        h = (dx)//(60*60)
        hx = (dx)%(60*60)
        m = (hx)//60
        mx = hx%60
        s = mx
        return f'{d} days {h} hr {m} min {s} sec'
```

```python
#4
class UserLoginInfo:
    def __init__(self,UserName,old_passwords):
        self.UserName = UserName
        self.old_passwords = [old_passwords,]

    def RetrievePassword(self):
        return self.old_passwords[-1]

    def ChangePassword(self,New_Password):
        if New_Password[0].isupper() and len(New_Password) > 7
        and New_Password.isalnum():
            if New_Password in self.old_passwords:
                return 'Password already used'
            self.old_passwords.append(New_Password)
            return 'Password updated successfully'
        return 'Invalid password'

    def Login(self,UserName,Password):
        if self.UserName == UserName and self.old_passwords[-1] == Password:
            return f'Welcome {UserName}'
        return 'Username or Password incorrect'
```