```python
# -*- coding: utf-8 -*-
"""
Éditeur de Spyder
"""

#3/ Boucles et fonctions: Applications

#%% 1. Opérations de bases

#1.1
def f11(a):
    return a.imag%2==0

#1.2
def f12(b):
    return type(b)==int

#%% 2. Ecrire un programme qui vous demade un entier et affiche la table de multiplication de cet entier de 1 à 10
def f2(a):
    for i in range (1,11):
        print("{}x{}={}".format(i,a,i*a))
    return

#%% 3. Le refuge
def refuge():
    L=[]
    for i in range (1,6):
        a=int(input("nbre personne dans refuge {} : ".format(i)))
        L.append(a)
    mini = L[0]
    pos = 0
    for i in range (1,len(L)):
        if mini > L[i]:
            mini = L[i]
            pos = i
    return "le refuge numero {} contient {} personnes !".format(pos+1,mini)

#%% 4. Les villages
def villages():
    p=int(input("Position actuelle : "))
    L=[]
    for i in range (1,6):
        L.append(int(input("positions villages {} : ".format(i))))
    nbre=0
    for i in range (len(L)):
        if abs(L[i]-p) <= 30:
            nbre+=1
    return "Vous pouvez atteindre {} villages".format(nbre)

def villages_bis():
    p=int(input("Position actuelle : "))
    nvil=int(input("nombre de villages a proximite :"))
    L=[]
    for i in range (1,nvil+1):
        L.append(int(input("positions villages {} : ".format(i))))
    nbre=0
    for i in range (len(L)):
        if abs(L[i]-p) <= 30:
            nbre+=1
    return "Vous pouvez atteindre {} villages".format(nbre)

#%% 5. La pyramide
def pyramide():
    bloc=int(input("nombre de blocs : "))
    bloc_ini=bloc
    i=0
    while bloc-(i+1)**2 >=0:
        i+=1
        bloc=bloc-i**2
    return "Hauteur : {} , Blocs utilises : {}".format(i,bloc_ini-bloc)

```

```python
#%% 6. Le rectangle
def rectangle(x,a,b):
    for i in range (b):
        print(x*a)
    return

#%% 7. Le cavalier
def cavalier(x,y):
    mvt=[[2,1],[1,2],[-1,2],[-2,1],[-2,-1],[-1,-2],[1,-2],[2,-1]]
    pos=[]
    for i in range(len(mvt)):
        if mvt[i][0]+x>=0 and mvt[i][0]+x<=7 and mvt[i][1]+y>=0 and mvt[i][1]+y<=7:
            pos.append([mvt[i][0]+x,mvt[i][1]+y])
    return pos

#%% 8. Le cavalier et le pion
def cavalier_pion(xc,yc,xp,yp):
    pos=cavalier(xc,yc)
    for i in range (len(pos)):
        if pos[i][0]==xp and pos[i][1]==yp:
            return "Possible"
    return "Impossible"


```