```
ï»¿#############################################################################
############################ The tower ################################
#############################################################################
import os
import keyboard
import time

class Player:
    def __init__(self):
        self.x=0
        self.y=0
        self.objects=[]


class Question:
    def __init__(self):
        self.pos=[]
        self.question=''
        self.answer=''
        self.object=''
        self.coordinateobj=[]


def screen_schema(n):
    for i in range(n):
        if schema[i][0]=='':
            break
        for j in range(n):

            if (schema[i][j]!=0 and schema!=''):

                print(schema[i][j],end='')


            elif schema[i][j]!='':
                    print ("\n")
                    break


def read_schema(nfile):

    fileschema='sch'+str(nfile)+'/schema.txt'
    with open(fileschema) as f:
        maps=f.read()
    contalinee=0
    contacolonne=0
    for elemento in maps:
        if elemento!='\n':
            schema[contalinee][contacolonne]=elemento
            if elemento=='@':
                hero.x=contacolonne
                hero.y=contalinee

        else:
            contacolonne=-1
            contalinee+=1

        contacolonne+=1
def read_solution(nfile):

    fileschema='sch'+str(nfile)+'/endschema.txt'
    with open(fileschema) as f:
        maps=f.read()
```

```
    contalinee=0
    contacolonne=0
    for elemento in maps:
        if elemento!='\n':
            schemafinale[contalinee][contacolonne]=elemento


        else:
            contacolonne=-1
            contalinee+=1

        contacolonne+=1


def question_screening(nfile):

    lista=[]
    fileschema='sch'+str(nfile)+'/questions.txt'
    q='*'
    with open(fileschema) as f:
        while q!='':
            q=f.readline()
            if q!='':
                lista.append(q)
                questions.append(Question())
    #lista=q.split(':')
    # print (lista)
    # print(questions)
    for (i,elemento) in enumerate(questions):
        segment=lista[i].split(':')
        # print(segment,i)
        for (j,info) in enumerate(segment):
            #print (i,j,info)
            if j==0:
                table=info.split(',')
                #print(i)
                questions[i].pos=[int(table[0]),int(table[1])]
            if j==1:
                table=info

                questions[i].question=table
            if j==2:
                table=info

                questions[i].answer=table
            if j==3:
                table=info

                questions[i].object=table
            if j==4:
                table=info.split(',')

                questions[i].coordinateobj=[int(table[0]),int(table[1])]


def check_solution(n):
    donot=False
    for i in range(n):
        for j in range(n):
            if schema[i][j]!=schemafinale[i][j]:
                donot=True
    #print (schemafinale)
    return donot
```

```
def presentation():
    fileschema='initialpage.txt'
    with open(fileschema) as f:
        page=f.read()
    print(page)

codelist=['cbgh','awer','prepo','minu','abrad','ulip','piol','jagbag','ulagu','birbx','ulbra','poyty','grbs']

nfile=1
start=1
nschemi=len(codelist)+1
#verde = '\u001b[92m'
#rosso = '\u001b[31m'
os.system('cls')
print(codelist)
presentation()
print('Inserisci codice avventura:=')
codice=input('')
for (v,elemento) in enumerate(codelist):
    if elemento==codice:
        start=v+1


for s in range(start,nschemi):
    nfile=s
    n=60
    schema=[[0 for i in range(n)] for j in range(n)]
    schemafinale=[[0 for i in range(n)] for j in range(n)]
    hero=Player()
    val=read_schema(nfile)
    read_solution(nfile)
    #print(val)
    #print (schema)
    loop=True
    key=''
    mem='-'
    os.system('cls')
    screen_schema(n)
    questions=[]
    labels=['1','2','3','4','5','6','7','8','9']
    question_screening(nfile)
    '''
    for elemento in questions:
        print (elemento.pos)
        print (elemento.question)
        print (elemento.answer)
        print (elemento.object)
        print (elemento.coordinateobj)

    '''
    #time.sleep(4)

    while loop:

        keyboard.press_and_release('backspace')

        time.sleep(0.1)
        if keyboard.is_pressed("q"):
            print(codelist[s-1])
            exit()
        if keyboard.is_pressed("up arrow"):
            if (schema[hero.y-1][hero.x]!='*') and (schema[hero.y-1][hero.x]!='X') and (schema[hero.y-1][hero.x]!='^') and
```

```
(schema[hero.y-1][hero.x]!='<') and (schema[hero.y-1][hero.x]!='>')and (schema[hero.y-1][hero.x]!='v'):
            schema[hero.y][hero.x]=mem
            mem=schema[hero.y-1][hero.x]
            schema[hero.y-1][hero.x]='@'

            hero.y-=1
            os.system('cls')
            screen_schema(n)

    if keyboard.is_pressed("down arrow"):

        if (schema[hero.y+1][hero.x]!='*') and (schema[hero.y+1][hero.x]!='X') and (schema[hero.y+1][hero.x]!='^')
and (schema[hero.y+1][hero.x]!='<') and (schema[hero.y+1][hero.x]!='>')and (schema[hero.y+1][hero.x]!='v') :

            schema[hero.y][hero.x]=mem
            mem=schema[hero.y+1][hero.x]
            schema[hero.y+1][hero.x]='@'

            hero.y+=1
            os.system('cls')
            screen_schema(n)

    if keyboard.is_pressed("right arrow"):
        if (schema[hero.y][hero.x+1]!='*') and (schema[hero.y][hero.x+1]!='X') and (schema[hero.y][hero.x+1]!='^')
and (schema[hero.y][hero.x+1]!='<') and (schema[hero.y][hero.x+1]!='>')and (schema[hero.y][hero.x+1]!='v'):

            schema[hero.y][hero.x]=mem
            mem=schema[hero.y][hero.x+1]
            schema[hero.y][hero.x+1]='@'

            hero.x+=1
            os.system('cls')
            screen_schema(n)

    if keyboard.is_pressed("left arrow"):

        if (schema[hero.y][hero.x-1]!='*') and (schema[hero.y][hero.x-1]!='X') and (schema[hero.y][hero.x-1]!='^') and
(schema[hero.y][hero.x-1]!='<') and (schema[hero.y][hero.x-1]!='>')and (schema[hero.y][hero.x-1]!='v'):

            schema[hero.y][hero.x]=mem
            mem=schema[hero.y][hero.x-1]
            schema[hero.y][hero.x-1]='@'

            hero.x-=1
            os.system('cls')
            screen_schema(n)
    if keyboard.is_pressed('h'):
        fileschema='help.txt'
        with open(fileschema) as f:
            page=f.read()

        print(page)


    if keyboard.is_pressed("c"):

        keyboard.press_and_release('backspace')
        if mem>='A' and mem<='Z':
            if len(hero.objects)<9:
                hero.objects.append(mem)
                mem='-'
        if mem=='?':
```

```
        for elemento in questions:
            if elemento.pos[0]==hero.x and elemento.pos[1]==hero.y:

                answer=input(str(elemento.question))
                #print (answer)
                if answer==str(elemento.answer):
                    #print('ok giusto!',elemento.object)
                    #print (elemento.coordinateobj[0],elemento.coordinateobj[1])
                    if schema[int(elemento.coordinateobj[1])][int(elemento.coordinateobj[0])]!='O':
                        schema[int(elemento.coordinateobj[1])][int(elemento.coordinateobj[0])]=elemento.object
                        elemento.answer='sasfjsdfkowqkr2'
    if mem=='g':
        #print('check')
        if check_solution(n)==False:
            print('You done!')
            loop=False

if keyboard.is_pressed("d") and mem=='-':
    os.system('cls')
    screen_schema(n)
    print (labels)
    print (hero.objects)
    choice=-1
    keyboard.press_and_release('backspace')
    while choice==-1:
        if keyboard.is_pressed("0"):
            choice=-2
        if keyboard.is_pressed("1"):
            choice=1
        if keyboard.is_pressed("2"):
            choice=2
        if keyboard.is_pressed("3"):
            choice=3
        if keyboard.is_pressed("4"):
            choice=4
        if keyboard.is_pressed("5"):
            choice=5
        if keyboard.is_pressed("6"):
            choice=6
        if keyboard.is_pressed("7"):
            choice=7
        if keyboard.is_pressed("8"):
            choice=8
        if keyboard.is_pressed("9"):
            choice=9

    if choice>0:
        if len(hero.objects)>=choice:
            schema[hero.y][hero.x]=hero.objects[choice-1]
            mem=hero.objects[choice-1]
            hero.objects.pop(choice-1)
```