Entrega de la primera práctica de AS2, que abarca ejercicios en lenguaje Python y se ejecutan sobre un SO Linux

Administración de Sistemas 2

Roberto Jiménez. 780906
Alberto Pérez 779212

```python
import os, sys, subprocess
import platform
import os.path

"""
Ejercicio 1a. Desarrollar un script donde se muestren los ficheros pasados por
parametro, asi como los permisos asociados a cada uno de ellos.
Sintaxis: listado.py [Dir]
Dir sera opcional de forma que si el usuario no lo inserta el script buscara
en el directorio actual
El script debera capturar todas las posibles excepciones como pueden ser:
a. Debe permitir el paso de parámetros con una sintaxis correcta
b. Si se produce alguna excepción grave a la hora de la búsqueda en sí, el
script debera comunicarlo.
"""

"""
Usage: Defining a method which obtains permissions of the files passed by
parameter
Name of method: checkFilePermissions
Date of creation: 18/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 20/02/2021
Parameters:
    Entry:  None
    Out: None
Documentation:  https://docs.python.org/3/library/subprocess.html
"""

def checkFilePermissions():

    contadorParametros = 1
    while contadorParametros < len(sys.argv):
        try:
            if (not os.path.isdir(sys.argv[contadorParametros])  and
               not os.path.isfile(sys.argv[contadorParametros])):
               #Check if the path is wrong
               print("Error in path: " + sys.argv[contadorParametros])
            else:
               subprocess.run(['ls', '-l', sys.argv[contadorParametros]], check =
True)
        except subprocess.CalledProcessError as err:
            print('Error in processor call:', err)
        except Exception as e:
            print('Error detected: ', e)
        contadorParametros += 1


"""
Usage: Defining main method of program
Name of method: main
Date of creation: 18/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 20/02/2021
Parameters:
    Entry:  None

Documentation:  https://docs.python.org/3/library/subprocess.html
"""
def main():
```

```
60
61      if  platform.system() != 'Linux':
62          #Check OS System
63
64          print("Error, the OS is not a UNIX machine. Getting out...")
65          exit(1)
66      if len(sys.argv) == 1:
67          #Case when no parameters are passed to the script, shows files and directory
    permissiones
68          os.system("ls -l")
69      else:
70          checkFilePermissions()
71
72
73
74  if __name__ == "__main__":
75      """
76      De esta forma, se comprueba en python para que al ejecutarse el script
77      vaya a la función indicada en este caso main.
78      """
79      main()
80
```

```python
import os, sys, subprocess
import platform
import os.path, stat

from os import listdir


"""
Ejercicio 2a. Modificar el anterior script para que se le pida al usuario por
consola el directorio donde tendrá que buscar y deberá mostrar todos
los ficheros del directorio, así como de los posibles subdirectorios.
Sintaxis: listado.py
El script deberá capturar todas las posibles excepciones como pueden ser:

a. Si el directorio que el usuario introduce no existe o no es un directorio,
deberá dar un error.
b. Si el usuario que lo invoca no tiene permisos de lectura del directorio,
el script deberá gestionar dicha excepción.
"""


"""
Usage: Defining a method that lists the files in a directory
Name of method: listDirectory
Date of creation: 20/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 20/02/2021
Parameters:
    Entry:
        - path: Directory from which the list of files or directories
           that exist is obtained
    Out:
        - List with files and directories
"""
def listDirectory(path):
    if os.access(path, os.R_OK):
        return os.listdir(path)
    return ["Error, directory has not reading permissions"]



"""
Usage: Definition of a recursive method that goes through all the directories and
subdirectories based on the directory passed by parameter
Name of method: recursivelyList
Date of creation: 20/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 20/02/2021
Parameters:
    Entry:
        - directory: Directory which is going to be listed
    Out: None

"""
def recursivelyList(directory):
    auxDirectories = []
    files = listDirectory(directory)
    print("\x1b[1;34m" + directory + ":")
    for file in files:
        path = directory + "/" + file
```

```
61          if(os.path.isdir(path)):
62              print("\x1b[1;36m" + file + " ", end="")
63              auxDirectories.append(path)
64          else:
65              print("\x1b[1;37m" + file + " ", end="")
66
67      print("\x1b[1;37m \n")
68      #
69      for directory in auxDirectories:
70          recursivelyList(directory)
71
72
73  """
74  Definition of the main method of the program
75  Name of method: main
76  Date of creation: 20/02/2021
77  Members: Roberto Jiménez y Alberto Pérez
78  Last modification: 20/02/2021
79  Parameters:  None, parameters are read by console
80  """
81  def main():
82
83      if  platform.system() != 'Linux':
84          #Check if is a UNIX machine
85          print("Error, the OS is not a UNIX machine. Getting out...")
86          exit(1)
87      if len(sys.argv) == 1:
88          #We chek that no parameters are passed
89          path = input('Write the path of the directory ')
90
91          if os.path.isdir(path):
92              print("")
93              recursivelyList(path)
94              #if os.chmod(path, )
95              #subprocess.run(['ls', '-l', '-R', path])
96
97      else:
98          #Case when parameters are passed to the script
99          print("Error, the parameters are not necessary")
100
101
102
103 if __name__ == "__main__":
104     """
105     In that case, we check that the program starts in the right function, in our
106     case, main()
107     """
108     main()
109
```

```python
import os, sys, platform
import psutil, time, smtplib, signal
from email.mime.text import MIMEText



"""
Ejercicio 3. Desarrollar un script que nos muestre por consola el consumo de CPU
y calcule internamente si dicho consumo supera el 70% 4 veces consecutivas,
en un periodo de tiempo preestablecido por el administrador.
En caso de producirse dicho evento, el script deberá enviar un mail al correo
del administrador
"""

"""
Usage: Makes the login and returns the serverSMTP
Name of method: create_session
Date of creation: 25/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 25/02/2021
Parameters:
    Entry:
        - emisor: Address of the account
        - passwd: The password of the mail
    Out:
        - serverSMTP, for sending the mail
"""

def create_session(emisor, passwd):
    #Connection with SMTP server
    serverSMTP = smtplib.SMTP('smtp.gmail.com',587)
    serverSMTP.starttls()

    #Log-in
    serverSMTP.login(emisor, passwd)
    return serverSMTP


"""
Generation of the mail that will be sended
Name of method: generateMail
Date of creation: 25/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 25/02/2021
Parameters:
    Entry:
        - mailFrom: Address of the account that the mail will be sended from
        - to: Address of the administrator mail
        - subject: Subject of the mail
        - content: Contet of the mail
    Out:
        - message, mail generated propertly
"""
def generateMail(mailFrom,to,subject, content):
    #Generation of the mail with MIMEText
    message = MIMEText(content, "plain")
    message["From"] = mailFrom
    message["To"] = to
    message["Subject"] = subject
    return message
```

```
61
62
63 """
64 Definition of the main method of the program
65 Name of method: main
66 Date of creation: 25/02/2021
67 Members: Roberto Jiménez y Alberto Pérez
68 Last modification: 25/02/2021
69 Parameters:  None, parameters are not needed
70 """
71 def main():
72     if  platform.system() != 'Linux':
73         #Check if is a UNIX machine
74         print("Error, the OS is not a UNIX machine. Getting out...")
75         exit(1)
76     if len(sys.argv) == 1:
77
78         #Declaration of the correct parameters for the mail
79         mailFrom = "roberatecaads2@gmail.com"
80         passwd='administracion2'
81         mailTo = "rooobertrl@gmail.com"
82
83         content = "Last 60 seconds, CPU has reached 4 times more than 70%"
84         subject = "High CPU!"
85
86         #Start an alarm every 60 seconds
87         signal.alarm(60)
88         reachedTimes = 0
89
90         #Checking when the cpu raises the 70 percent 4 times in a row
91         while(True):
92             CPUpercent = psutil.cpu_percent(interval=3)
93             if CPUpercent >= 70:
94                 reachedTimes += 1
95             else:
96                 reachedTimes = 0
97             if reachedTimes == 4:
98                 try:
99                     smtp = create_session(mailFrom,passwd)
100                    email = generateMail(mailFrom, mailTo, subject,content )
101                    smtp.sendmail(mailFrom, mailTo, email.as_string())
102                    smtp.quit()
103                except Exception as e:
104                    print("Excepcion: ", e)
105                break
106
107    else:
108        #Case when parameters are passed to the script
109        print("Error, the parameters are not necessary")
110
111
112 if __name__ == "__main__":
113     """
114     In that case, we check that the program starts in the right function, in our
115     case, main()
116     """
117     main()
118
```

```python
import imaplib, pyzmail, platform, sys, datetime, locale

"""
Ejercicio 4. Desarrollar un script en Python que elimine de una cuenta de correo de Gmail los
emails no leídos en un intervalo de fechas previamente preestablecido.
Nota: paquetes necesarios para el uso de las acciones requeridas (imapclient,pyzmail)
"""

"""
Definition of the main method of the program
Name of method: main
Date of creation: 27/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification:
    - 3/03/2021 If a date is wrong, exit from the program
    - 5/03/2021 Calling deleteMails for delete all incorrect mails
Parameters:  None, parameters are read by console
"""
def main():

    if  platform.system() != 'Linux':
        #Check if is a UNIX machine
        print("Error, the OS is not a UNIX machine. Getting out...")
        exit(1)
    if len(sys.argv) == 1:
        #Gets the initial date and check if its correct
        startDate = input("Put the start date: (dd-mm-yyyy): ")
        startDateAux = checkDate(startDate)
        if(not(startDateAux)):
            print("Getting out...")
            exit()

        #Gets the initial date and check if its correct
        finalDate = input("Put the final date: (dd-mm-yyyy): ")
        finalDateAux = checkDate(finalDate)
        if(not(finalDateAux)):
            print("Getting out...")
            exit()

        #Check if the range has sense
        if(finalDateAux < startDateAux):
            print("Error, start date is bigger than final date")
            print("Getting out...")
            exit()

        #Making login in the mail
        server = imaplib.IMAP4_SSL('imap.gmail.com')
        server.login('roberatecaads2@gmail.com', 'administracion2')
        select_info = server.select("INBOX")

        #Deleting all mails
        deleteMails(server, startDateAux, finalDateAux)

    else:
        #Case when parameters are passed to the script
        print("Error, the parameters are not necessary")


"""
```

```
Checks if the date passed got by console is correct
Name of method: checkDate
Date of creation: 28/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification:
    - 3/03/2021. Checks the format of the date
    - 5/03/2021. Checks if the date is correct
Parameters:
    Entry:
        - date: String of the date provided by the user
    Out:
        - datetime, the date in datetime format
"""
def checkDate(date):
    try:
        dateAux = str(date).split("-")
        return datetime.datetime(int(dateAux[2]),int(dateAux[1]),int(dateAux[0]))
    except ValueError:
        print("Error, date doesn't exists")
        return None
    except IndexError:
        print("Error, incorrect date format")
        return None



"""
Delete all mails in a date range
Name of method: deleteMails
Date of creation: 27/02/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification:
    - 27/02/2021 Delete all mails without date range
    - 28/02/2021 Using the since sentence for delete the mails in the date range
Parameters:
    Entry:
        - server: Connection with the mail provider
        - startDate: Initial date provided by the user
        - finalDate: Final date provided by the user
    Out:
"""
def deleteMails(server,startDate,finalDate):
    query = ("UNSEEN SINCE " + startDate.strftime("%d-%b-%Y") + " BEFORE "
             + finalDate.strftime("%d-%b-%Y"))
    estado, messages = server.search(None,query)
    messages = messages[0].split()

    for mail in messages:
        server.store(mail, '+FLAGS', '\\Deleted')
    print(str(len(messages)) + " mail deleted")
    server.expunge()


if __name__ == "__main__":
    """
    In that case, we check that the program starts in the right function, in our
    case, main()
    """
    main()

```

```python
import os, sys, platform

"""
Ejercicio 5. Desarrollar un script que nos muestre por consola información relevante del
identificador de proceso que el usuario inserte por consola.
Sintaxis: proceso.py
El script deberá mostrar la siguiente información del proceso: identificador UID, propietario
PID, PPID, Prioridad absoluta ”C”, Prioridad Relativa “PRI” y la Dirección en memoria o en
disco del proceso “ADDR”
"""

"""
Definition of the main method of the program
Name of method: main
Date of creation: 7/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 7/03/2021
Parameters:  None, parameters are passed
"""
def main():
    if  platform.system() != 'Linux':
        #Check if is a UNIX machine
        print("Error, the OS is not a UNIX machine. Getting out...")
        exit(1)

    if len(sys.argv) == 1:
        #Case when parameters are not passed to the script
        print("Error, a process id is necessary")
    elif len(sys.argv) == 2:
        query = "ps -o uid,pid,ppid,c,pri,addr " + sys.argv[1]
        os.system(query)
    else:
        #Case when parameters are passed to the script
        print("Error, more parameters than necessaries")


if __name__ == "__main__":
    """
    In that case, we check that the program starts in the right function, in our
    case, main()
    """
    main()

```

```python
import os, platform, sys,subprocess

"""
Ejercicio 6. Desarrollar un script para que busque a aquellos procesos con una prioridad relativa
superior a 11 y les decremente su prioridad en 5 unidades; posteriormente deberá mostrar que
procesos han sido susceptibles de modificación junto con la nueva prioridad asignada.
"""

"""
Definition of the main method of the program
Nombre: main
Fecha de creacion: 8/03/2021
Miembros: Roberto Jiménez y Alberto Pérez
Última modificación: 8/03/2021
Parámetros: Parameters are passed
"""
def main():

    if  platform.system() != 'Linux':
        #Check if is a UNIX machine
        print("Error, the OS is not a UNIX machine. Getting out...")
        exit(1)
    if len(sys.argv) == 1:

        #Getting the PID list of processes
        vectorPID = subprocess.getoutput("ps -l | awk '$8 <= 0 && $4 {print $4}'")
        lista = vectorPID.split()
        vectorNI=subprocess.getoutput("ps -l| awk '$8>=0 && $4 {print $8}'")
        lista2 = vectorNI.split()
        lista2.remove("NI")

        #Updates the proccesses priority
        counter = 0
        for pid in lista:
            if int(lista2[counter]) >= 0:
                query = "sudo renice -n " + str(int(lista2[counter]) - 5) + " -p " + pid
                os.system(query)
                print("The process " + pid + " has been modified")
            else:
                print("No process with priority bigger than ten")
            counter += 1
    else:
        #Case when parameters are passed to the script
        print("Error, the parameters are not necessary")



if __name__ == "__main__":
    """
    De esta forma, se comprueba en python para que al ejecutarse el script
    vaya a la función indicada en este caso main.
    """
    main()

```

```python
import os, platform, sys,subprocess, copy, filecmp, time

"""
Ejercicio 7. Desarrollar un script que detecte ficheros duplicados en un directorio (por ejemplo
el dicrectorio /tmp) y automáticamente los elimine de dicho directorio
"""

"""
Definition of the main method of the program
Nombre: main
Fecha de creacion: 11/03/2021
Miembros: Roberto Jiménez y Alberto Pérez
Última modificación: 11/03/2021
Parámetros: Parameters are passed in the calling process
"""
def main():
    if  platform.system() != 'Linux':
        #Check if is a UNIX machineputt
        print("Error, the OS is not a UNIX machine. Getting out...")
        exit(1)
    if len(sys.argv) == 1:
        #Case when parameters are not passed to the script
        directory = input("Your current directory is: " + os.getcwd() + " please insert another"+
                " directory to compare: " + '\n' )
        if(os.path.isdir(directory)):
            deleteDuplicateFiles(os.getcwd(), directory)
        else:
            print("Error, the passed directory does not exist")
    elif  len(sys.argv) == 2:
        #Case when a parameter is passed to the script
        directory = input("Directory passed:" + sys.argv[1] + " please insert another"+
                " directory to compare: " + '\n' )
        if(os.path.isdir(directory)):
            deleteDuplicateFiles(sys.argv[1], directory)
        else:
            print("Error, one of the directories does not exist")
    elif len(sys.argv) == 3:
        #Case when 2 parameters are passed to the script

        if(os.path.isdir(sys.argv[1]) and os.path.isdir(sys.argv[2])):
            deleteDuplicateFiles(sys.argv[1], sys.argv[2])
        else:
            print("Error, one of the directories does not exist")

    else:
        print("Error, the parameters are not necessary")




"""
Deletes the duplicate files
Nombre: deleteDuplicateFiles
Fecha de creacion: 11/03/2021
Miembros: Roberto Jiménez y Alberto Pérez
Última modificación: 11/03/2021
Parámetros:
```

```
    Entry:
        - dir1: First directory for getting the files
        - dir2: Second directory for getting the files
    Out:
"""
def deleteDuplicateFiles(dir1, dir2):
    filesDir1 = subprocess.getoutput("ls " + dir1).split() #Get a list of files from directory1
    filesDir2 = subprocess.getoutput("ls " + dir2).split() #Get a list of files from directory2


    for file1 in filesDir1:
        for file2 in filesDir2:
            #Compares if the name of two files is the same
            if file1 == file2:
                pathFile1 = dir1 + "/" + file1
                pathFile2 = dir2 + "/" + file2

                #Compares if the content of two files is the same
                if filecmp.cmp(pathFile1, pathFile2, shallow=False):
                    dateTimeFile1 = (time.ctime(os.path.getctime(pathFile1)), pathFile1)
                    dateTimeFile2 = (time.ctime(os.path.getctime(pathFile2)), pathFile2)
                    borrar = max(dateTimeFile1[0], dateTimeFile2[0])
                    #Deletes the newest file
                    if(borrar == dateTimeFile1[0]):
                        os.system("rm " + dateTimeFile1[1])
                        print("File: " + dateTimeFile1[1] + " deleted")
                    elif(borrar == dateTimeFile2[0]):
                        os.system("rm " + dateTimeFile2[1])
                        print("File: " + dateTimeFile2[1] + " deleted")


if __name__ == "__main__":
    """
    De esta forma, se comprueba en python para que al ejecutarse el script
    vaya a la función indicada en este caso main.
    """
    main()

```

```python
import os, sys

"""
Ejercicio 8. Generar un script que nos permita generar ficheros tar, inicialmente el script nos
mostrará un menú con las siguientes opciones:
a. Generacion fichero tar
b. Extracción fichero tar
c. Visualización de la información del fichero tar (nombre del fichero, propietario, y
tamaño)
d. Listado de todos los archivos incluidos en el fichero tar
e. Validación de que el fichero tar se ha generado correctamente, para ello el menú deberá
solicitar previamente el fichero tar a validar.
Sintaxis: generatar.py
"""


"""
Usage: Defining a method that create a Tar file from an input wrote by user
Name of method: optionA
Date of creation: 16/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 16/03/2021
Parameters:
    Entry: None
    Out: None, output it's tar file
"""
def optionA():
    print("Name of new tar file: ")
    fileTarName = input()
    next = "y"
    #Loop for adding more files
    while next == "y":
        print("File name to introduce into a tar file: ")
        fileName = input()
        query = "tar -rvf " + fileTarName + " " + fileName
        os.system(query)
        print("Do you want add more files (y/n)?")
        next = input()


"""
Usage: Defining a method that extracts files from a Tar file
Name of method: optionB
Date of creation: 16/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 16/03/2021
Parameters:
    Entry: None
    Out: None, output is the files extracted
"""
def optionB():
    print("Name of the .tar file")
    fileTarName = input()
    os.system(("tar -xvf" + fileTarName))

"""
Usage: Defining a method that lists info from a Tar file
```

```
Name of method: optionC
Date of creation: 16/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 16/03/2021
Parameters:
    Entry: None
    Out: None, output only is showed by terminal
"""
def optionC():
    print("Name of .tar file")
    fileTarName = input()
    os.system(("ls -l " + fileTarName))


"""
Usage: Defining a method that lists the files in a Tar file
Name of method: optionD
Date of creation: 16/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 16/03/2021
Parameters:
    Entry: None
    Out: None, output only is showed by terminal
"""
def optionD():
    print("Name of .tar file: ")
    fileTarName = input()
    os.system(("tar -tf " + fileTarName))


"""
Usage: Defining a main method of program, which is able to work with Tar files,
differents options are splitted in some methods (OptionA, B, C and D).
Here user select from a menu differents options
Name of method: main
Date of creation: 14/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 16/03/2021
Parameters:
    Entry: None
    Out: None, info is requested by console
"""
def main():
    continuar="y"
    while continuar == "y":
        print("------------------------------------------------------------ \n"+
                "a) Generate tar file \n" +
                "b) Extract tar file \n" +
                "c) View info about tar file \n" +
                "d) List all files from a tar file\n" +
                "e) Exit \n" +
                "------------------------------------------------------------")
        print("$ ", end ="")
        option = input()
        if option == "a":
            optionA()
        elif option == "b":
            optionB()
        elif option == "c":
            optionC()
```

```
118         elif option == "d":
119             optionD()
120         elif option == "e":
121             exit(0)
122         else:
123             print("Incorrect option")
124 
125 
126         print("Do you want to do another operation (y/n)")
127         continuar = input()
128 
129 
130 if __name__ == "__main__":
131     """
132     De esta forma, se comprueba en python para que al ejecutarse el script
133     vaya a la función indicada en este caso main.
134     """
135     main()
136 
```

```python
import os, platform, sys,subprocess


"""
Ejercicio 8. Realizar un script que genere el alta de cuatro usuarios en el sistema, dichos
usuarios deberán pertenecer al grupo Alumnos, para ello se tendrán que realizar una serie de
etapas:
- Necesarias :
Editar el fichero /etc/passwd para definir la nueva cuenta de usuario.
Introducir una contraseña inicial (comando passwd).
Crear el directorio propio (HOME) del usuario.
- Conveniente para el usuario :
Copiar los ficheros de arranque por defecto al directorio del usuario.
Crear el directorio de correo del usuario y definir alias del correo.
Realizar las tareas de configuración especificas a algunas aplicaciones.
Informar al usuario cuando la apertura haya sido realizada.
- Conveniente para el administrador :
Añadir el usuario al fichero /etc/group.
Registrar información contable.
Introducir al usuario en la base de datos de usuarios de la entidad.
Configurar las cuotas de disco.
Verificar que la cuenta ha sido creada correctamente.
"""


"""
Usage: Defining a method that create a user into a Unix system
Name of method: createUser
Date of creation: 20/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 20/03/2021
Parameters:
    Entry:
        - nameUser: String which contains user's name
        - group: String which contains group name for new user
    Out: None
"""
def createUser(nameUser, group):
    userExists = os.system("sudo useradd -g " + group + " "+  nameUser)
    if(userExists != 0):
        return
    os.system("sudo passwd " + nameUser)
    os.system("sudo mkdir /home/" + nameUser)
    os.system("sudo chown " + nameUser + ":" + group + " -R /home/" + nameUser)
    os.system("sudo usermod -s /bin/bash " + nameUser)
    copyStartFiles()
    print("User has been created correctly")


"""
Usage: Defining a method that configure start files for new user, files copied
are:
    - ~/.bashrc
    - ~/.home
    - ~/.bash_logout
Name of method: copyStartFiles
Date of creation: 20/03/2021
Members: Roberto Jiménez y Alberto Pérez
```

```
Last modification: 20/03/2021
Parameters:
    Entry: None
    Out: None
Documentation: https://www.sever-world.info/en/note?os=Ubuntu_18.04&p=mail&f=3
"""
def copyStartFiles():
    #Start files are copied to new user directory
    os.system("sudo cp ~/.bashrc /home/" + nameUser)
    os.system("sudo chown " +  nameUser + " /home/" + nameUser + "/.bashrc" )

    os.system("sudo cp ~/.profile /home/" + nameUser)
    os.system("sudo chown " +  nameUser + " /home/" + nameUser + "/.profile" )

    os.system("sudo cp ~/.bash_logout /home/" + nameUser)
    os.system("sudo chown " +  nameUser + " /home/" + nameUser + "/.bash_logout")




"""
Usage: Defining a main method
Name of method: main
Date of creation: 20/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 20/03/2021
Parameters:
    Entry: None
    Out: None
"""
def main():

    if  platform.system() != 'Linux':
        #Check if is a UNIX machine
        print("Error, the OS is not a UNIX machine. Getting out...")
        exit(1)
    if len(sys.argv) == 1:
        for i in range(4):
            user = input("Introduce username for new user ")
            createUser(user, "Alumnos")

    else:
        #Case when parameters are passed to the script
        print("Error, the parameters are not necessary")



if __name__ == "__main__":
    """
    De esta forma, se comprueba en python para que al ejecutarse el script
    vaya a la función indicada en este caso main.
    """
    main()

```

```python
import os, sys, random

"""
Ejercicio 9. Realizar un script que nos genere password cifrados de 10 caracteres aleatorios,
siendo dichos caracteres los que a continuación se describen dentro de un vector:
VECTOR="0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwx
yz"
Posteriormente realizar la compresión de un fichero de texto:
casa bbb zzzzz ff  casa 3b 5z ff (2f)
Y a continuación realizar un cifrado simple de cada uno de los caracteres:
Carácter cifrado = (Código ASCII(carácter) + número constante) Módulo 256
¿Cómo se podría desencriptar?
Nota: se puede usar el valor devuelto por la variable RANDOM
"""


"""
Usage: Defining a main method that generate a ciffer key using ASCII code and
module operation
Name of method: main
Date of creation: 24/03/2021
Members: Roberto Jiménez y Alberto Pérez
Last modification: 24/03/2021
Parameters:
    Entry: None, Parameters are requested by console
    Out: None, key is shown by console
"""
def main():
    #Declare a variable  which contains characters permitted
    array="0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"

    randomNumber = random.randrange(3000)
    key=""
    cifferValue =""

    for i in range (10):
        char = random.randrange(len(array)) + 1
        key += array[(char - 1):char:1]
        cifferValue += str((ord(key[i:i + 1: 1]) + randomNumber) % 256) + " "

    print("The key is: " + key)
    print("The ciffer key is:" + cifferValue)
    print("The random number is: " + str(randomNumber))




if __name__ == "__main__":
    """
    De esta forma, se comprueba en python para que al ejecutarse el script
    vaya a la función indicada en este caso main.
    """
    main()

```