|  |  |
| --- | --- |
| **Ex. 9** | **EXPLORING NESTED LISTS** |
| **Date:25/03/14** |  |

/

**Aim:**

To explore nested lists in Python by writing programs for the following:

1. Obtain 2 matrices from a user in the form of nested lists and perform matrix multiplication and addition, without using built in functions.
2. Given the names and grades for each student in a class, store them in a nested list and print the name(s) of any student(s) having the second lowest grade.
3. Check if a given list contains nested lists.
4. Find the list in a list of lists whose sum of elements is the highest.

**Algorithm:**

**(a)**

STEP 1: Prompt for matrices A and B.

STEP 2: Set up a zero matrix for the product if dimensions match.

STEP 3: Check dimension compatibility.

STEP 4: Perform matrix multiplication if possible.

STEP 5: Print error if dimensions don't match.

STEP 6: Display the resulting matrix**.**

**(b)**

STEP 1: Prompt for the number of students and initialize lists for students, grades, and lowest grades.

STEP 2: Collect names and grades of students, storing each as a sublist in a list.

STEP 3: Extract grades from the list and sort them.

STEP 4: Identify the second lowest grade in the sorted list.

STEP 5: Find all students with the second lowest grade and store their details.

STEP 6: Print the names and grades of students with the second lowest grade.

**(c)**

STEP 1: Prompt the user to input a list and store it.

STEP 2: Define a function to check for nested lists within the provided list.

STEP 3: Iterate through each item in the list; if any item is a list, return True.

STEP 4: If the function returns True, print that the list contains a nested list.

STEP 5: If no nested list is found, print that the list does not contain any nested lists.

**(d)**

STEP 1: Prompt the user to input a list of lists and store it.\STEP 2: Initialize a variable to keep track of the highest sum and an empty list to store the sublist with the highest sum.

STEP 3: Iterate through each sublist in the list.

STEP 4: Calculate the sum of each sublist.

STEP 5: Compare the sum of the current sublist with the highest recorded sum.

STEP 6: If the current sum is greater, update the highest sum and replace the sublist in the final list.

STEP 7: Print the sublist with the highest sum and the value of the highest sum.

**Program:**

**(a)**

a=eval(input("Enter the elements in matrix A:"))

b=eval(input("Enter the elements in matrix B:"))

prod = [[0 for \_ in range(len(b[0]))] for \_ in range(len(a))]

if(len(a)==len(b[0])):

  for i in range(len(a)):

     for j in range(len(b[0])):

        for k in range(len(b)):

            prod[i][j]+=a[i][k]\*b[k][j]

else:

   print("Multipliaction is not possible")

   for p in prod:

       print(p)

**(b)**

n=int(input("Enter the number of students:"))

lst=[]

index=[]

low=[]

for i in range(n):

    name=input("Enter the name:")

    grade=int(input("Enter the grade"))

    lst.append([name,grade])

for x in lst:

    index.append(x[-1])

index.sort()

lowest=index[1]

for x in lst:

   if lowest==x[-1]:

       low.append(x)

print("The second lowest grade=",low)

**(c)**

lst1=eval(input("Enter the list"))

def check(lst1):

    for x in lst1:

       if(isinstance(x,list)):

          return True

       return False

if check(lst1):

   print("It contains nested list")

else:

   print("It doesn't")

**(d)**

lst=eval(input("Enter the elements in the list"))

final=[0]

highest=0

for x in lst:

  sum=0

  for i in x:

     sum+=i

if sum>highest:

    highest=sum

    final.pop()

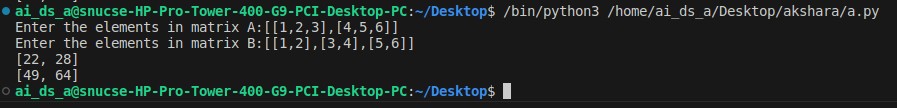
    final.append(x)

print(final)

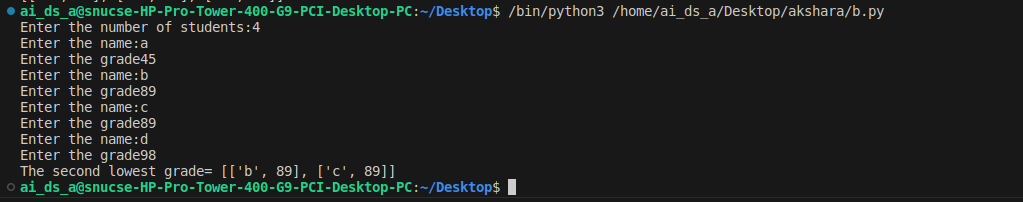
print(highest)

**Screenshot of Output:**

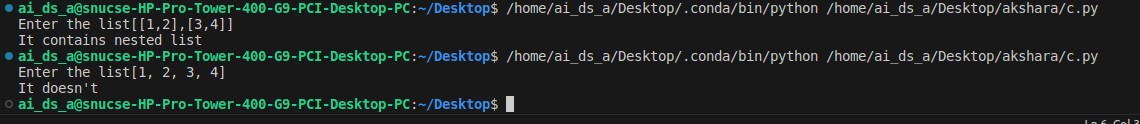
(a)



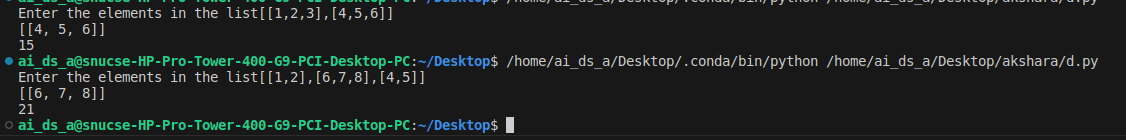
(b)



(c)



(d)



**Result:**

Thus, programs have been written and executed to explore nested lists in Python.