ASPC-A 2023 Day 5 (PM) Problem Set Solutions

This is the **solutions manual** for the **problem sets for Day 5 (PM)**.

This solutions manual has **9 problems**. The items in the problem list below are links to each problem. You can click a problem name to jump to its page in the PDF file.

ASPC-A 2023 Day 5 (PM) Problem Set Solutions

Solution for Seriously, Is It Sorted?

Get the input as a **a list of integers** and then print the integers in reverse order.

Solution:

```
N = int(input())
data = [int(i) for i in input().split()]
data.sort()
for x in range(len(data)-1,-1,-1):
      print(data[x])
```

ASPC-A 2023 Day 5 (PM) Problem Set Solutions

Solution for Data Plan

Get the integer T as input then loop T times.

At each iteration, get a line as input and then output its length.

Solution:

```
T = int(input())
for t in range(1,T+1):
      line = input()
      print("Line "+str(t)+" is "+str(len(line))+" characters long.")
```

ASPC-A 2023 Day 5 (PM) Problem Set Solutions

Solution for Find The Median

We get the input and print out the numbers as needed.

Since the input is on **separate lines**, we use a loop to get input.

Solution:

```
N = int(input())
data = [float(i) for i in input().split()]
data.sort()
if len(data)%2 == 1:
     print("%.4f" % data[len(data)//2])
else:
     median = (data[len(data)//2] + data[len(data)//2-1])/2
     print("%.4f" % median)
```

ASPC-A 2023 Day 5 (PM) Problem Set Solutions

Solution for ASCII GUI

Since there are only 5 lines of input, you can manually get all five lines.

To output 20 dashes, simply **copy-paste from the problem set**.

Solution:

```
L1 = input()
L2 = input()
L3 = input()
L4 = input()
L5 = input()

print( "--------------------" );
print( "5:" + L5 );
print( "4:" + L4 );
print( "3:" + L3 );
print( "2:" + L2 );
print( "1:" + L1 );
print( "--------------------" );
```

Solution for Mean Ranges

This is a straightforward item, made easier using math library functions.

Solution:

```
N = int(input())
data = [int(i) for i in input().split()]

sum = 0
for x in data:
      sum += x

print( str(max(data)-min(data))+" %.2f" % (sum/N))
```

Solution for Dictionary Time

Sort the input, and store it separately. Then check if the input and the sorted lists are equal.

Solution:

```
N = int(input())
data = []
for x in range(N):
	data.append( input() )

sortedVersion = sorted(data)

answer = True
for x in range(N):
	if sortedVersion[x] != data[x]:
		answer = False

if answer:
	print("yes")
else:
	print("no")
```

Solution for Nonogram Hints

You can convert the periods into spaces, and then split the string.

Solution:

```
inputNums = [int(i) for i in input().split()]
N = inputNums[0]
M = inputNums[1]
rowData = input().split()
hints = rowData[:M]
inputrow = rowData[M]

spacesrow = ""
for i in range(len(inputrow)):
     if inputrow[i] == ".":
          spacesrow = spacesrow + " "
     if inputrow[i] == "#":
          spacesrow = spacesrow + "#"

row = spacesrow.split()

valid = True
if len(hints)!=len(row):
     valid = False
if valid:
     for i in range(len(hints)):
          if int(hints[i]) != len(row[i]):
               valid = False

if valid:
     print("Valid!")
else:
     print("Invalid.")
```

ASPC-A 2023 Day 5 (PM) Problem Set Solutions

Solution for Top Target

Sorting the input groups equal elements together.

You can then count the sizes of each group by looping.

Solution:

```
N = int(input())
data = []
for i in range(N):
      data.append(input())

data.sort();

topFrequency = 0
topString = ""

currFrequency = 0
currString = ""

for i in range(N):
      if data[i] == currString:
            currFrequency += 1
      else:
            currString = data[i]
            currFrequency = 1

      if currFrequency > topFrequency:
            topFrequency = currFrequency
            topString = currString

print("TOP TARGET: " + str(topString))
print("APPEARS " + str(topFrequency) + " TIME(S)")
```

ASPC-A 2023 Day 5 (PM) Problem Set Solutions

Solution for Panel Puzzle

Create functions to rotate a grid by 90 degrees, and check if one grid fits in the other grid.

Solution:

```
def rotate90( grid, N ):
      T = []
      for i in range(N):
            T.append([])
            for j in range(N):
                  T[i].append(grid[N-j-1][i])
      return T

def isValid( A, B, N ):
      valid = True
      for i in range(N):
            for j in range(N):
                  if B[i][j]=='#' and A[i][j]!='.':
                        valid = False
      return valid

N = int(input())
A = []
for i in range(N):
      A.append(input())
B = []
for i in range(N):
      B.append(input())

v1 = isValid(A,B,N)
B = rotate90(B,N)
v2 = isValid(A,B,N)
B = rotate90(B,N)
v3 = isValid(A,B,N)
B = rotate90(B,N)
v4 = isValid(A,B,N)

if v1 or v2 or v3 or v4:
      print("Yes")
else:
      print("No")
```