Algorithmics
Correction Final Exam #1 (P1)

Undergraduate 1^{st} year (s1) – Epita

3 Jan. 2017 - 10 : 00

Solution 1 **(Stacks and garage... –** ***3 points***)

1. (a) Are the following sequences valid?
   i. Yes!
   ii. No: The fourth exit is done on an empty garage (stack), indeed v_1,v_2 et v_3 are already out.

   (b) **The rule:**
   A sequence containing `'E1'`, `'E2'` and `'D'` is *admissible* if it contains as `'D'` as cumulated `'E1'` and `'E2'` and if all its actions can be accomplished in the specified order of the sequence: we can only pop if there is at least one element. At the end, the stack (garage) has to be empty.

Solution 2 **(Binary Search: search "path" –** ***2 points*****)**

The sequences ② et ③ are impossible:

① 46, we go on the right - 65, we go on the right - 81, we go on the left - 73, we go on the left - 70, we go on the left - **66**

② 31, we go on the right - **62, we go on the right** - 90, we go on the left - 72, we go on the left - **61 cannot be here, it is not higher than 62**

③ 36, we go on the right - 70, we go on the left - **53, we go on the right - 50, cannot be here, it is not higher than 53**

④ 35, we go on the right - 51, we go on the right - 55 we go on the right - 58, we go on the right - 61, we go on the right - **66**

Solution 3 **(Test - *1 point*)**

`test(`x`, `L`)` tests whether x is in the list L.

Solution 4 **(Integers $\leftrightarrow$ list – *5 points*)**

1. The function `int_to_list(`n`, `p`)` returns the list of the p digits of n:

```
def int_to_list(n, p):
    L = []
    while n != 0:
        L.append(n % 10)
        n = n // 10
    for i in range(p-len(L)):
        L.append(0)
    return L

#-------------------------------------------------------------------------

def int_to_list2(n, p):
    L = []
    while p > 0:
        L.append(n % 10)
        n = n // 10
        p -= 1
    return L
```

2. The function `list_to_ints(`$[d_1, d_2, \cdots, d_p]$`)` returns the pair of integers $(d_1 d_2 \cdots d_p,\ d_p \cdots d_2 d_1)$:

```
def list_to_ints(L):
    left = 0
    right = 0
    n = len(L)
    for i in range(n):
        left = left * 10 + L[i]
        right = right * 10 + L[n-i-1]
    return (left, right)

#-------------------------------------------------------------------------

def list_to_ints2(L):
    left = 0
    right = 0
    p = 1
    for i in range(len(L)):
        left = left * 10 + L[i]
        right = right + L[i]*p
        p = p * 10
    return (left, right)
```

Solution 5 **(Histogram and sort – *4 points*)**

1. The function `hist(`L`)` returns the list that represents the histogram of the values in L (L contains only digits):

```
def hist(L):
    H = []
    for i in range(10):
        H.append(0)
    # ou H = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
    for e in L:
        H[e] += 1
    return H
```

2. The function `sort(`L`)` returns the list L sorted in increasing order (L contains only digits):

```
def sort(L):
    H = hist(L)
    L = []
    for i in range(10):
        for nb in range(H[i]):
            L.append(i)
    return L
```

Solution 6 **(Kaprekar – *5 points*)**

The function `Kaprekar(`n`, `p`)` performs the Kaprekar routine to the positive p-digit integer n, till it reaches the same value twice. It displays the computed values.

```
def Kaprekar(n, p):

    L = []

    while not test(n, L):
        print(n, end=' -> ')

        L.append(n)

        digits = int_to_list(n,p)
        digits = sort(digits)
        (low, high) = list_to_ints(digits)
        n = high - low

    print(n)
```