# lab2

October 16, 2020

## 1 Setup

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

## 2 Dataset Preparation

### 2.1 Data Generation

[2]:
```
n=30
a = np.random.randint(0,n,size=n)
a
```

[2]:
```
array([ 8, 18, 14, 25,  3,  1,  3,  3,  8, 22, 15,  4, 28, 19, 13, 27, 14,
       11, 22,  1, 10, 17,  0,  4, 18, 17, 28, 28,  7, 25])
```

### 2.2 Auxilary Array Definition

- dp: dp[i] represents the length of the longest subsequence with the last element i
- pre_node: pre_node[i] represents the previous node of element i in the longest subsequence with the last element i
- path: path[i] records each path for the longest subsequence with the last element i

[3]:
```
dp = [1]*n
pre_node = [i for i in range(n)]
path = [[] for _ in range(n)]
```

## 3 Calculate dp (without recursion)

[4]:
```
for j in range(1,n):
    for i in range(j):
        if a[i]<=a[j] and dp[j]<dp[i]+1:
            dp[j]=dp[i]+1
            pre_node[j]=i
```

[5]:
```
print(dp)
print(pre_node)
```

```
[1, 2, 2, 3, 1, 1, 2, 3, 4, 5, 5, 4, 6, 6, 5, 7, 6, 5, 7, 2, 5, 7, 1, 5, 8, 8,
9, 10, 6, 9]
[0, 0, 0, 1, 4, 5, 4, 6, 7, 8, 8, 7, 9, 10, 8, 13, 14, 8, 13, 5, 8, 16, 22, 11,
21, 21, 24, 26, 23, 24]
```

## 4 Use list pre_node to calculate list path

[6]:
```
for j in range(n):
    if pre_node[j]==j:
        path[j].append(j)
        continue
    tmp = path[pre_node[j]].copy()
    tmp.append(j)
    path[j]=tmp
```

[7]:
```
path
```

[7]:
```
[[0],
 [0, 1],
 [0, 2],
 [0, 1, 3],
 [4],
 [5],
 [4, 6],
 [4, 6, 7],
 [4, 6, 7, 8],
 [4, 6, 7, 8, 9],
 [4, 6, 7, 8, 10],
 [4, 6, 7, 11],
 [4, 6, 7, 8, 9, 12],
 [4, 6, 7, 8, 10, 13],
 [4, 6, 7, 8, 14],
 [4, 6, 7, 8, 10, 13, 15],
 [4, 6, 7, 8, 14, 16],
```

```
 [4, 6, 7, 8, 17],
 [4, 6, 7, 8, 10, 13, 18],
 [5, 19],
 [4, 6, 7, 8, 20],
 [4, 6, 7, 8, 14, 16, 21],
 [22],
 [4, 6, 7, 11, 23],
 [4, 6, 7, 8, 14, 16, 21, 24],
 [4, 6, 7, 8, 14, 16, 21, 25],
 [4, 6, 7, 8, 14, 16, 21, 24, 26],
 [4, 6, 7, 8, 14, 16, 21, 24, 26, 27],
 [4, 6, 7, 11, 23, 28],
 [4, 6, 7, 8, 14, 16, 21, 24, 29]]
```

## 5 Get the longest subsequence and visualize it

[8]:
```
pos = -1
length = -1
for i in range(n):
    if dp[i]>length:
        pos = i
        length=dp[i]
length,path[pos]
```

[8]:
```
(10, [4, 6, 7, 8, 14, 16, 21, 24, 26, 27])
```

[9]:
```
rest = list(set(range(n))-set(path[pos]))
num = len(rest)

plt.figure(figsize=(10, 5))

plt.grid('minor')

plt.xlim(-1,n+3)
plt.ylim(-1,max(a)+8)
plt.scatter(path[pos],np.array(a)[path[pos]],c='orange',label='The Points in the␣
 ↪Longest Subsequence')
plt.scatter(rest,np.array(a)[rest],c='blue',label='The Rest Points in the␣
 ↪Initial Sequence')
plt.plot(path[pos],np.array(a)[path[pos]],c='red',linestyle='-.
 ↪',label=None,alpha=0.9)
plt.legend(loc=2)
plt.xlabel('The position of each point')
plt.ylabel('The value of each point')
```

```
plt.title('The Longest Increasing Subsequence')
plt.show()
plt.savefig("path.png",dpi=300)
```