Q1 [**Bash**]

Write a bash script which takes an XML file describing a directory subtree as input (as command line argument) and creates a subtree under the current directory. Note that file has two properties, name and size (in bytes). You need to create files with specified name and size.
Example:

Sample1.xml

```
<dir>
        <name> Level1 </name>
        <file>
                <name> f1 </name>
                <size> 10240 </size>
        </file>
        <dir>
                <name> Level12 </name>
                <file>
                        <name> f12 </name>
                        <size> 2048 </size>
                </file>
                <file>
                        <name> f13 </name>
                        <size> 4096 </size>
                </file>
                <dir>
                        <name> level22</name>
                </dir>
        </dir>
</dir>
<dir>
        <name> src </name>
        <file>
                <name> hello.c </name>
                <size> 654 </size>
        </file>
</dir>
```

```
$ q1.sh sample1.xml
$ tree
    Level1
      |----------------f1
      |----------------Level12
                       |------------------f12
                       |------------------f13
                       |----------------Level22
    src
     |----------------hello.c
```

Notes:

(1) Assume the input XML file is syntactically correct listing files and directories.
(2) The levels of nesting can be of any depth.
(3) The size of the files created should match the specification exactly

Q2 [**awk**]

The "ps" command displays information on active processes in your system. Do "man ps" for more information.

Each column in the output corresponds to a metric. The following command

ps -eLo uname,pid,ppid,lwp,sz,rss,start_time,time,cmd

gives information on all the active processes with columns for username (uname column), memory usage (SZ column), CPU time (TIME column), PID (system wide unique process ID), LWP (system wide unique thread ID) and some more.

The processes have a unique process ID (PID) and each run on behalf of a user (uname). Each process can also have multiple threads. Thus multiple threads with a unique thread ID (LWP) will have the same process ID (PID). And these processes come under a user (uname). Since all the threads of a given process share the memory, the memory usage indicated for each thread is the total memory usage of the entire process. The CPU time indicated for each thread of a given process can be added to represent the CPU time of the given process (considering its active threads only).

From this information, given the output of *ps* command with necessary columns, create a report on:
1) the number of users
2) the number of processes for each username
3) the total number of threads for each username
4) the total CPU consumption for each username
5) the total Memory consumption for each username

The "ps" input given to your awk script may not have its columns in a fixed order. You need to parse the first row to detect the column order. For simplifying the delimiter assume that the "cmd" column is always at the end.

Example test runs:

```
ps -eLo uname,ppid,pid,lwp,start_time,sz,time,rss,cmd > order_1.txt
ps -eLo uname,pid,ppid,lwp,sz,rss,start_time,time,cmd > order_2.txt
awk -f filename.awk order_1.txt
awk -f filename.awk order_2.txt
```

Q3 [**Python**]
Given an array, print the contents in a **balanced binary search tree** format. Balanced Binary Search Tree is a Binary Search Tree where at each node the height of its children subtrees differ by at most 1.

Note-

a) If we try to read the tree from left to right (without concerning the height), the numbers should appear in **ascending** order.
b) You are expected to only output the numbers in Balanced Binary Search Tree Format and not create the tree explicitly. So prior knowledge of Binary Search Trees are not required.
c) You are allowed to use only O(1) extra space. (i.e, you can only keep one copy of the input)
d) You are not allowed to use any Graph related python library. Please try to solve it using list, dictionary and/or numpy arrays.

Example -
Input - [1,2,3,4,5,6,7]
Output -

```
    4
  2   6
1 3 5 7
```

Input - [5,6,2,1,9]
Output -

```
      5
  1       6
    2       9
```

Input - [9,8,7,6,5,4,3,2,1]
Output -

```
          5
    2             7
 1     3       6     8
         4               9
```