RBAC Configuration:
Input:
Number_of_Users: int
Number_of_Roles: int
Number_of_Permissions: int
Depth_of_RH: int
Nature_of_RH: int //0: Stanford, 1: Hybrid
R_Connectivity: int //0: random, 1: uniform
U_Connectivity: int //0: random, 1: uniform, 2: exactly 1
P_Connectivity: int //0: random, 1: uniform, 2: exactly 1
Connectivity: int

Algorithm:

1. We create three objects: *users*, *roles*, *permissions*. Each object size is determined by *Number_of_Users*, *Number_of_Roles*, and *Number_of_Permissions*, respectively, and each object stores the id of users, roles, and permissions, respectively.
2. We calculate *Number_of_Roles*/*Depth_of_RH* which returns an int. This tells us how many roles we should assign for each layer in the role hierarchy
3. We read the Connectivity value:
    a. If Depth_of_RH > 1
        i. If R_Connectivity = 0
            1. If Nature_of_RH = 0
                a. Each role at a specific layer is assigned to roles at a layer directly below it randomly
            2. Else if Nature_of_RH = 1
                a. Each role at a specific layer is assigned to roles at any layer below it randomly
        ii. Else if R_Connectivity = 1
            1. If Nature_of_RH = 0
                a. Each role at a specific layer is assigned to roles at a layer directly below it uniformly
            2. Else if Nature_of_RH = 1
                a. Each role at a specific layer is assigned to roles at any layer below it uniformly
        iii. If U_Connectivity = 0
            1. If Nature_of_RH = 0
                a. Each user is assigned random roles which are at the higest layer.
            2. Else if Nature_of_RH = 1
                a. Each user is assigned random roles which are at any layer.
        iv. Else if U_Connectivity = 1
            1. If Nature_of_RH = 0

a. Each user is assigned uniform roles which are at the higest layer.
2. Else if Nature_of_RH = 1
a. Each user is assigned uniform roles which are at any layer.

v. Else if U_Connectivity = 2
1. If Nature_of_RH = 0
a. Each user is assigned to exactly 1 role which is at the higest layer.
2. Else if Nature_of_RH = 1
a. Each user is assigned to exactly 1 role which is at any layer.

vi. If P_Connectivity = 0
1. If Nature_of_RH = 0
a. Each role at the lowest layer is assigned random permissions
2. Else if Nature_of_RH = 1
a. Each role at any layer is assigned random permissions

vii. Else if P_Connectivity = 1
1. If Nature_of_RH = 0
a. Each role at the lowest layer is assigned uniform permissions
2. Else if Nature_of_RH = 1
a. Each role at any layer is assigned uniform permissions

viii. Else if P_Connectivity = 2
1. If Nature_of_RH = 0
a. Each role at the lowest layer is assigned to exactly one permission
2. Else if Nature_of_RH = 1
a. Each role at any layer is assigned to exactly one permission

b. Else if Depth_of_RH = 1
i. If U_Connectivity = 0 and P_Connectivity = 0
1. Each user is assigned random roles and each role is assigned random permissions

ii. Else if U_Connectivity = 1 and P_Connectivity = 1
1. Each user is assigned uniform roles and each role is assigned uniform permissions

iii. Else if U_Connectivity = 2 and P_Connectivity = 2
1. Each user is assigned to exactly 1 role and each role is assigned to exactly one permission

Output:
RBAC configuration in the following format:

```
#UA
<User> <Assigned set of roles>
#PA
<Role> <Assigned set of permissions>
#RH
<Role> <Assigned set of roles>
```

Session Profile:
Input:
RBAC Configuration output
Number_of_Sessions: int
Number_of_Sessions_per_AccessCheck: int
Number_of_Roles_per_Session: int
Nature_of_Roles: int
Number_of_AccessChecks: int
Nature_of_AccessChecks: int

Algorithm:

1- CreateSession:
   a. If *Number_of_Sessions_per_AccessCheck* is reached, call AccessCheck.
   b. If total sessions exceed *Number_of_Sessions_per_AccessCheck*, delete one of the previous sessions created
   c. Activate a number of roles matching *Number_of_Roles_per_Session*.
      i. If *Nature_of_Roles* is 0, activate only roles that share the same permissions
      ii. If *Nature_of_Roles* is 1, activate roles at different levels of role hierarchy
      iii. If *Nature_of_Roles* is 2, activate some junior roles but not all, making sure to not activate parent roles which would activate other junior roles in order to provide DSoD
2- AccessCheck
   a. Perform a total number of access checks according to *Number_of_AccessChecks*
      i. If *Nature_of_AccessChecks* is 0, perform access checks to all permissions created
      ii. If *Nature_of_AccessChecks* is 1, perform access checks to allowed permissions
3- Repeat 1 and 2 until *Number_of_Sessions* is reached
4- Delete all remaining sessions

Output:
Session Profile output is of the following format:

i <item id> <set of roles>
a <item id> <set of permissions>
d <item id>

test_result1: 724, 723, 714, 713, 697, 696, 695, 661, 646, 639.
test_result2: 143, 142, 141, 134, 128, 89.