TDDD04 Exercises:

1. **White box testing**

```
1. public boolean checkDate(boolean [] availability, int start_time,
2. int duration){
3. Boolean can_schedule = true;
4. int i = 0;

5. if (start_time + duration > 18 | start_time < 8){
6. print("cannot schedule meeting outside of work hours");
7. can_schedule = false;
8. }

9. while (can_schedule & (i < duration)){
10.   if(availability[start_time + i - 8] = false){
11.   print("cannot schedule meeting because of time conflict");
12.   can_schedule = false;
13.   }


14.   if (can_schedule = true)
15.   print("meeting scheduled");

16.   return can_schedule;
17.   }
```

   a. For the code above calculate the cyclomatic complexity and provide the set of basis paths.
   b. Based on these basis paths, provide a set of test-cases, trying to choose values meaningfully.

2. **Integration and system testing**

```
1.  public int diceGame (int a, int b){

2.   if (beteween1and6(a) && beteween1and6(b))
3.     then {
4.      if(isSix(a)||isSix(b)) then return 20;
5.      else  points = a+b;
6.   }
7.   else {
8.        print ("Impossible input");
9.        points = -1;
10.    }
11.   return points;
12.  }
13. }
14. public boolean between1and6(int a){
15.  return (a<=6)&&(a>=1);
16. }

17. public boolean isSix(int a){
18.  return (a==6);
19. }
```

   a. For the code above build a flow graph representation, select an interesting test-case and provide the list of Module Executions Paths (MEPs)
   b. From the MEPs defined in a. build a MM-path graph