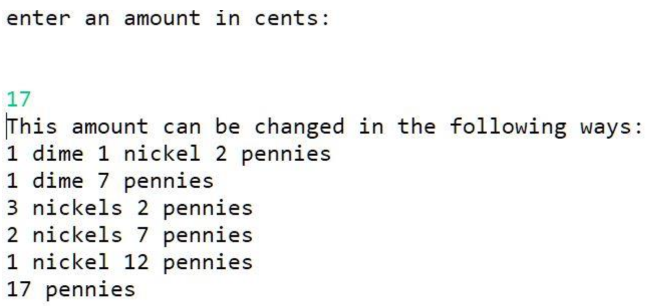
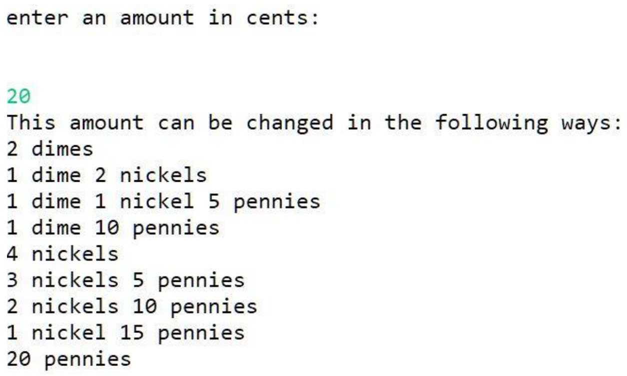
Rajwinder kaur
Student No.: 216907602
EECS2011 SECTION O

Question 1: Coins.java

Output for question one Coins.java



```
enter an amount in cents:


17
This amount can be changed in the following ways:
1 dime 1 nickel 2 pennies
1 dime 7 pennies
3 nickels 2 pennies
2 nickels 7 pennies
1 nickel 12 pennies
17 pennies
```

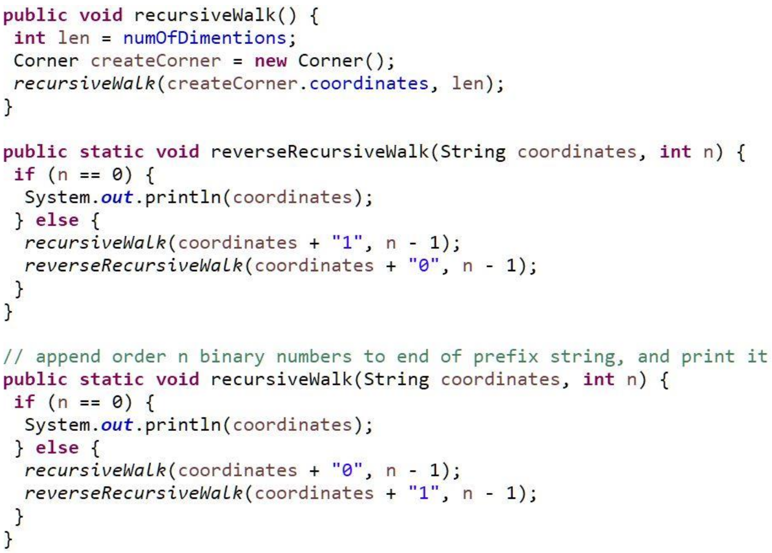


```
enter an amount in cents:


20
This amount can be changed in the following ways:
2 dimes
1 dime 2 nickels
1 dime 1 nickel 5 pennies
1 dime 10 pennies
4 nickels
3 nickels 5 pennies
2 nickels 10 pennies
1 nickel 15 pennies
20 pennies
```

This program prints all the different ways change can be made while checking the plurals/singular noun depending on the number of coins. In its recursive function it starts from the condition check at step 0 and then add 1 to quarter and keep on repeating the step till the amount is either 0 or goes less than 0 and at that point it moves to the next condition check step 1 provided amount became negative in the previous step. and then again repeats the same step with rest of the coins and condition check increases by 1 at every coin change.

Question 2 Hypercube.java

running time of part b:



```
public void recursiveWalk() {
 int len = numOfDimentions;
 Corner createCorner = new Corner();
 recursiveWalk(createCorner.coordinates, len);
}

public static void reverseRecursiveWalk(String coordinates, int n) {
 if (n == 0) {
  System.out.println(coordinates);
 } else {
  recursiveWalk(coordinates + "1", n - 1);
  reverseRecursiveWalk(coordinates + "0", n - 1);
 }
}

// append order n binary numbers to end of prefix string, and print it
public static void recursiveWalk(String coordinates, int n) {
 if (n == 0) {
  System.out.println(coordinates);
 } else {
  recursiveWalk(coordinates + "0", n - 1);
  reverseRecursiveWalk(coordinates + "1", n - 1);
 }
}
```

Time taken on my computer to run recursive algorithm for 3D hyper cube   = 0 milli second no. of corners generated $= 2^n$

Rajwinder kaur
Student No.: 216907602
EECS2011 SECTION O

when $n=1$, $T(1) = \{0,1\}$

$T(n) = C + T(n-1) + T(n+1)$

BY INDUCTION

$$T(n) = \begin{cases} T(n-1) + C & \text{if } n>0 \\ C & \text{if } n=0 \end{cases}$$

$$\begin{aligned} T(n) &= T(n-1) + C \\ &= T(n-1-1) + C + C \\ &= T(n-3) + 3C \\ &= T(n-4) + 4C \\ &\vdots \\ &= T(n-k) + kC \\ &= T(n-k) + C \end{aligned}$$

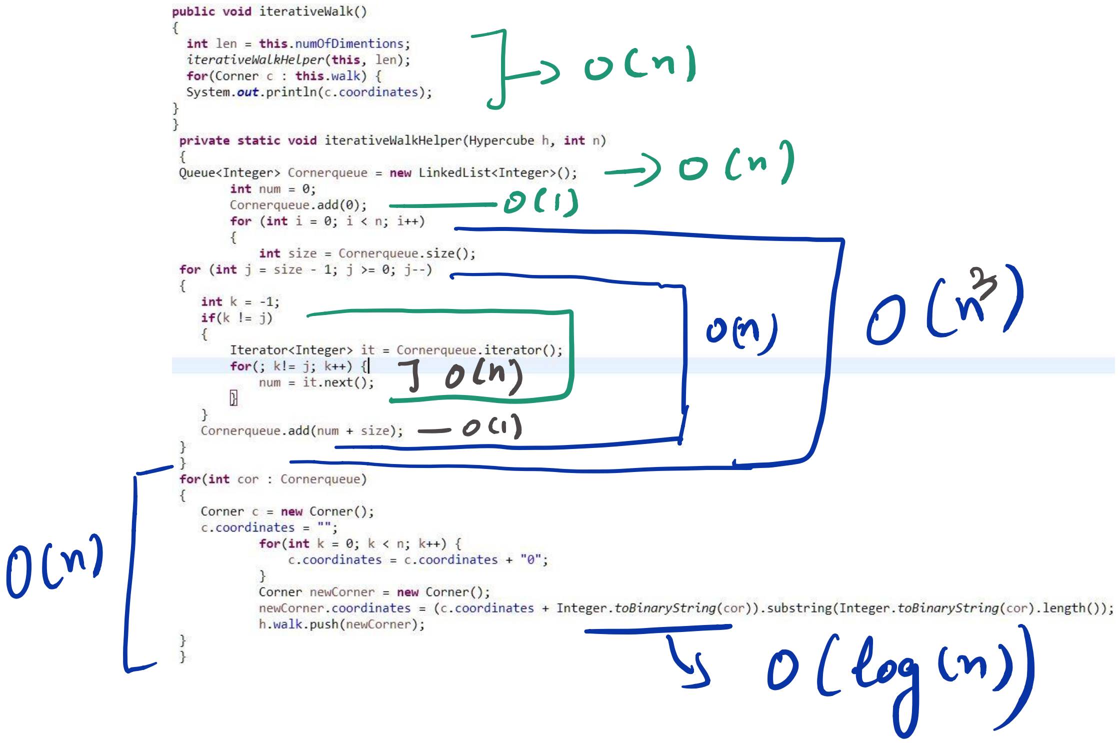
$T(n) = T(n-k) + C$

$\simeq O(n)$

So, it is O(n) for the recursive solution.

Rajwinder kaur
Student No.: 216907602
EECS2011 SECTION O

running time of part c:

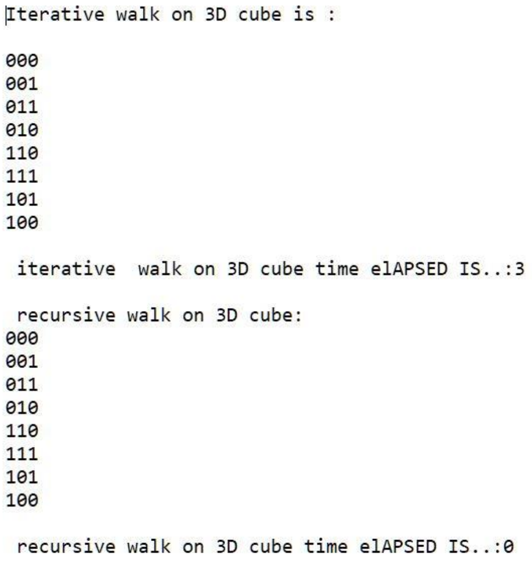


```java
public void iterativeWalk()
{
  int len = this.numOfDimentions;
  iterativeWalkHelper(this, len);
  for(Corner c : this.walk) {
  System.out.println(c.coordinates);
}
}
 private static void iterativeWalkHelper(Hypercube h, int n)
 {
 Queue<Integer> Cornerqueue = new LinkedList<Integer>();
        int num = 0;
        Cornerqueue.add(0);
        for (int i = 0; i < n; i++)
        {
            int size = Cornerqueue.size();
 for (int j = size - 1; j >= 0; j--)
 {
    int k = -1;
    if(k != j)
    {
        Iterator<Integer> it = Cornerqueue.iterator();
        for(; k!= j; k++) {
            num = it.next();
        }
    }
    Cornerqueue.add(num + size);
 }
 }
 for(int cor : Cornerqueue)
 {
    Corner c = new Corner();
    c.coordinates = "";
            for(int k = 0; k < n; k++) {
                c.coordinates = c.coordinates + "0";
            }
            Corner newCorner = new Corner();
            newCorner.coordinates = (c.coordinates + Integer.toBinaryString(cor)).substring(Integer.toBinaryString(cor).length());
            h.walk.push(newCorner);
 }
 }
```

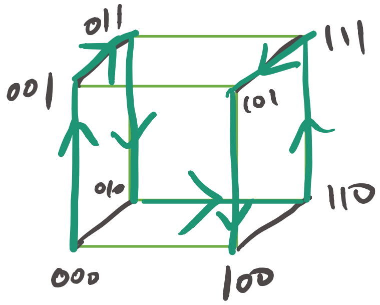
From the above analysis of the iterative function and other functions being used in it it turns out to be O(n) = n^3

Rajwinder kaur
Student No.: 216907602
EECS2011 SECTION O

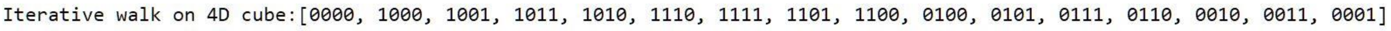
Result I/O of question 2 :



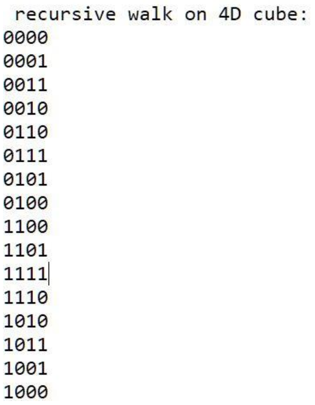
```
Iterative walk on 3D cube is :

000
001
011
010
110
111
101
100

 iterative  walk on 3D cube time elAPSED IS..:3

 recursive walk on 3D cube:
000
001
011
010
110
111
101
100

 recursive walk on 3D cube time elAPSED IS..:0
```



Every corner visted as per requirement .



```
Iterative walk on 4D cube:[0000, 1000, 1001, 1011, 1010, 1110, 1111, 1101, 1100, 0100, 0101, 0111, 0110, 0010, 0011, 0001]
```



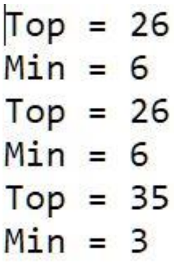
```
 recursive walk on 4D cube:
0000
0001
0011
0010
0110
0111
0101
0100
1100
1101
1111
1110
1010
1011
1001
1000
```

```
Iterative walk on 5D cube:[00000, 10000, 10001, 10011,
10010, 10110, 10111, 10101, 10100, 11100, 11101, 11111,
11110, 11010, 11011, 11001, 11000, 01000, 01001, 01011,
01010, 01110, 01111, 01101, 01100, 00100, 00101, 00111,
00110, 00010, 00011, 00001]

 recursive walk on5D cube:
0000
0001
```

```
0011
0010
0110
0111
0101
0100
1100
1101
1111
1110
1010
1011
1001
1000
```

Question 3 : ADT Stack.   AugStack.java

I/O of ADT is :



```
Top = 26
Min = 6
Top = 26
Min = 6
Top = 35
Min = 3
```

Solution of Question 3.