Tamar Harizy 209927128

Batzion YomTov 314955766

Homework Assignment #3

1) recDelete(L,x)

```
if (p = x)                          // p = Head(L)
        p = next(p)
        return
if (next(p) = x)
        next(p) = next(x)
        return
recDelete(next(p))

return
```

2) sameStack(s1, s2)

```
if (isEmpty(s1) AND isEmpty(s2)
        return true
else if (!isEmpty(s1) AND !isEmpty(s2)
        a = pop(s1)
        b = pop(s2)
        if (a == b)
                bool same = sameStack(s1, s2)
                push(s1, a)
                push(s2, b)
                if (same)
                        return true
                else
                        return false
        else
                push(s1, a)
                push(s2, b)
                return false
else
        return false
```

Tamar Harizy 209927128

Batzion YomTov 314955766

Homework Assignment #3

3)

komarachi (n)

```
if (n <= 3)
        return 1
if (n % 2 == 0)  // if n is even
        return komarachi(n-1) + komarachi (n-2)
else                        // n is odd
        return komarachi(n-1) + komarachi (n-2) + komarachi (n-3)
```

4)

1. 2 7 4 3 1 5 6 8 pivot 2
2. 2 1 4 3 7 5 6 8 pivot 2
3. 1 2 4 3 7 5 6 8 pivot 4
4. 1 2 3 4 7 5 6 8 pivot 7
5. 1 2 3 4 6 5 7 8 pivot 6
6. 1 2 3 4 5 6 7 8

5) 2, 3, 6, 8, 17, 4, 12, 5, 9, 13, 40, 0, 10

6) a.

isBST (T)

```
if (isEmpty(T))
        return true
 if (!emptyTree(T))
      visit (root (T))
        if (root(T) < right(T) AND root(T) > left(T))
                isBST (left(T))
                isBST (right(T))
        else
                return false
        return true
```

Tamar Harizy 209927128

Batzion YomTov 314955766

Homework Assignment #3

b. $\Theta(n)$

7) **LongestEvenPath (T)**

```
if (isEmpty(T))
        return 0
 if (!emptyTree(T))
        if (key(root(T))%2 == 0)                    // if the key is even
              return max (LongestEvenPath (letf(T)+1), LongestEvenPath(right(T)+1)
        else                                        // if the key is odd
               LongestEvenPath(left(T))
               LongestEvenPath(right(T))
               return 0
```

8) **countNodes**(T)

```
 If (isEmpty(T))
return 0
 return countNodes(left(T)) + countNodes(right(T)) + 1
```

Size(T)

```
If (isEmpty(T))
        Return NULL
Else
        s == countNodes(T)      // s is the size variable
        Key(root(T)) == s
        Return Size(right(root(T))
        Return Size(left(T))
```