CSE101 – Introduction to Programming

Tutorial 8 Solutions

1. Recursive:

```
def fibonacci (n):
        if (n= = 1):
                return 0
        elif (n= = 2):
                return 1
        return fib(n-1) + fib(n-2)
```

Iterative:

```
def fibonacci (n):
        if (n= = 1):
                return 0
        elif (n= = 2):
                return 1
        x=0
        y=1

        for i in range(3,n+1):
                z=x+y
                x=y
                y=z
        return y
```

2.
```
def linear_search(arr,low,high,x):
        if ( low>high):
                return -1
        elif ( arr[high]= = x):
                return high
        return linear_search(arr,low,high-1,x)
```

3.
```
def pascal(n):
  if n = = 1:


    return [1]
  else:
    line = [1]
    previous_line = pascal(n-1)
    print (previous_line)
    for i in range(len(previous_line)-1):
      line.append(previous_line[i] + previous_line[i+1])
    line += [1]
  return line


n=int(input())

print (pascal(n))
```

4.
```
 power(2,10) -> power(2,5) -> power(2,2) -> power(2,1) -> power(2,0)

x=2              x=2            x=2         x=2           x=2

n=10             n=5            n=2         n=1           n=0

a=32       <-    a=4      <-    a=2    <-   a=1     <-    return 1

return 1024   return 32      return 4    return 2
```

Note: Each call frame frees up memory after it's return statement is executed, that is,

From right to left, as per the above diagram.

5. **def countPairs(s):**

```
def countPairs(s):
        if (len(s)<3):
                return 0
        elif (s[0]= = s[2]):
                return 1 + countPairs(s[1:])
        return countPairs(s[1:])
```