1.

a) A random variable is a variable whose value is a numerical outcome of a random phenomenon.

b)

X	2	3	4	5	6	7	8	9	10	11	12
P(X)	1/36	2/36	3/36	4/36	5/36	6/36	5/36	4/36	3/36	2/36	1/36

c) The mean of a discrete random variable, X, is its weighted average, which is called the expected value of X.

The mean of the case is 2*1/36+3*2/36+4*3/36+⋯+12*1/36=7

If X is a discrete random variable with μ mean , then the variance of X is

$$\sigma_X^2 = (x_1 - \mu_X)^2 p_1 + (x_2 - \mu_X)^2 p_2 + \cdots + (x_k - \mu_X)^2 p_k$$
$$= \sum (x_i - \mu_X)^2 p_i$$

The standard deviation (σ_X) is the square root of the variance.

The standard deviation of this case is $\sqrt{(2-7)^2 * \frac{1}{36} + (3-7)^2 * \frac{2}{36} + \cdots + (12-7)^2 * \frac{1}{36}} = \frac{\sqrt{210}}{6} \approx 2.415$

d) The probability of two people entering restaurant at the same time is 1/36, The probability of three people entering restaurant at the same time is 2/36, the probability of four people entering restaurant at the same time is 3/36, and so on.

The mean of the case is 7 and the standard deviation is 2.415.

2.

```
for (int i = 1; i <= N; i++)
    for (int j = i; j <= N; j++)
        for (int k = j; k <= N; k++)
            count++;
```

A) In the worst case, the code in the 3-Sum triple loop will be executed $\sum_{i=1}^{N}\sum_{j=i}^{N}\sum_{k=j}^{N} 1$ times. So the time-complexity should be $c * \sum_{i=1}^{N}\sum_{j=i}^{N}\sum_{k=j}^{N} 1$ (c is the time for a single operation) which is the same as $O(N^3/6)$.

B) If it is a 2-Sum loop, it would be $c * \sum_{i=1}^{N}\sum_{j=i}^{N} 1$

which equals $(N-1)+(N-2)+\cdots+1=1/2*N^2-1/2N=O(N^2/2)$

3.

StackOfStrings(): create an empty stack

void push(String item): insert a new string onto stack

String pop(): remove and return the string most recently added

boolean isEmpty(): is the stack empty?

int size(): number of strings on the stack

4.
The code is shown in the java file.

5.
The code is shown in the java file:
The class “Main” is related to the class “LinkedList” and the class “student” which implements linked list to build the stack. The class “Main2” is related to the class “FixedCapacityStack” and the class “student” which implements fixed capacity stack to build the stack.
h) For both “LinkedList” and “FixedCapacityStack”, the time-complexity is O(1) for the methods “isEmpty()”, “push” and “pop”.
j) For fixed array size, when oversizing, use resizing array for array implementation. When undersizing, throw exception if pop from an empty stack.

6.
A) The code is resizing the array when oversizing happens. It will double the size of the stack.
B) The stack is fixed capacity, so it may face the problem of oversizing. Double the size of the stack is the most commonly accepted way to fix this problem in a not that expensive way.
C) I would shrink the array when calling the pop method to save the space. The code would be like the following:

```
public String pop() {
    String item = s[--N];
    s[N] = null;
    if (N > 0 && N == s.length/4) resize(s.length/2);
    return item;
}
```