VE 477 Homework6

Wang Yichao, ID: 517370910011

- **Exercise 1.** 1. By definition, there are n monomials, and each monomial corresponds to a permutation. We discuss in two cases. (a) If there is a perfect match, WLOG we can assume the elements on one diagonal line are variables X_i, j. All the rest are 0. Then $det(A)$ is the multiple of all these variables which is not zero. (b) If there is not a perfect match, then at least one node is not connected, which leads to a row in the matrix to be 0. This will immediately make $det(A)$ to be zero.

  2.

Algorithm 1 Lovasz

Input: Graph $G = (V, E)$ and its matrix A
Output: a boolean that indicates whether there is a perfect match

```
for each X_{i,j} in A do
    X_{i,j} = uniformly random from 1 to n^2
end for
if det(A) == 0 then
    return False
else
    return True
end if
```

  3. Time complexity would be $|V|^2$ since we need to set the random number for the whole matrix. Error probability would be $1 - \frac{1}{n}$ according to Schwartz-Zippel Lemma.

  4. This strategy can be faster than the deterministic algorithm, because it is easier to calculate a determinant with value than a determinant with variables(it can be an expensive n^2-variate polynomial of degree n). And the correct probability is quite high.

- **Exercise 2.** 1.

Algorithm 2 middle node

Input: HeadNode
Output: MiddleNode

```
MiddleNode = HeadNode
EndNode = HeadNode
while EndNode.next != NULL do
    MiddleNode = MiddleNode.next
    EndNode = EndNode.next.next
end whilereturn MiddleNode
```

  2.

Algorithm 3 cycle

Input: HeadNode
Output: A boolean indicate whether there is a cycle

```
slow = HeadNode
fast = HeadNode.next
while fast.next != NULL do
    if slow == fast then
        return True
    end if
    slow = slow.next
    fast = fast.next.next
end while
return False
```

Since we need to traverse the list, $\mathcal{O}(n)$ time complexity, and no extra space needed, so $\mathcal{O}(1)$ space complexity.

- **Exercise 3.** 1. Since n kinds of coupons, we need at least n boxes.

  2. The probability of getting a new couple given that we have already get $j-1$ couples is $\frac{n-j+1}{n}$. Then we calculate the reciprocal and get $E\left[X_j\right] = \frac{n}{n-j+1}$.

  3. It is super interesting that some clever genius can finish question 3 easily by skipping question 2.

  To calculate $E\left[X\right]$, we need to add up $E\left[X_j\right]$. $E\left[X\right] = p = \sum_{j=1}^{n} \frac{n}{n-j+1} = n \times \sum_{j=1}^{n} \frac{1}{j} = n(\log n + \gamma)$, where γ is the Euler constant and n goes to infinity. So we get $E\left[X\right] = \Theta(n \log n)$.

  4. The formula above shows that the number of boxes grows a little bit faster than the growth of coupon, which is linear. If the couples are identical and we only need to collect n pieces of coupons, that will be linear. So we can make more profit by introducing n different kinds of coupons.