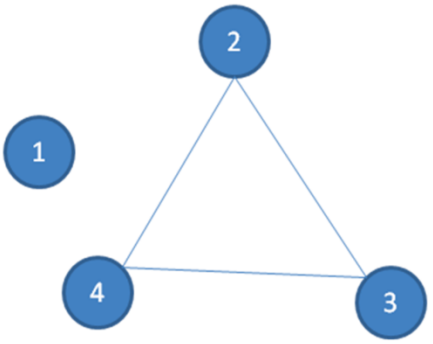
BÀI TẬP THỰC HÀNH

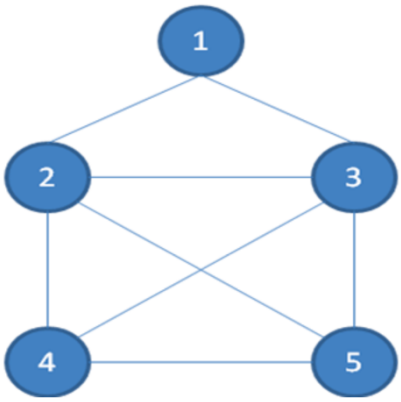
Cho đơn đồ thị G có thể là đồ thị có hướng hoặc vô hướng. Mỗi đồ thị đều có các thuộc tính: số đỉnh của đồ thị, ma trận kề/ma trận liên thuộc, …. Ngoài ra, mỗi đồ thị còn có các hành vi (là các phương thức được mô tả trong từng câu hỏi bên dưới). Áp dụng mẫu thiết kế sao cho phù hợp để hiện thực các yêu cầu sau:

Câu 1: (1) Viết phương thức đọc ma trận kề từ một file test.txt



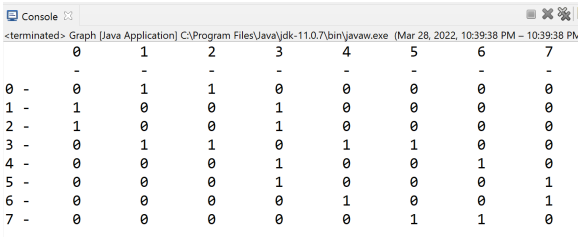

```
public boolean loadGraph(String pathFile) {
        // implement code
}
```

Câu 2: (1) Viết phương thức in ra ma trên kề của đồ thị




```
public void printMatrix(int[][] matrix) {
        // implement code
}

public void printMatrix() {
        // implement code
}
```



Câu 3: (1) Viết phương thức kiểm tra đồ thị có hợp lý không (có hướng hoặc vô hướng)

```
public boolean checkValid() {
        // implement code
}
```

Câu 4: (1) Viết phương thức kiểm tra đồ thị có phải là đồ thị vô hướng hay không

```
public boolean checkUnGraph(){
        // implement code
}
```

Câu 5: (1) Viết phương thức thêm một cạnh vào đồ thị

```
public void addEdge (int[][] matrix, int v1, int v2) {
        // implement code
}
```

Câu 6: (1) Viết phương thức xóa một cạnh vào đồ thị

```
public void removeEdge (int[][] matrix, int v1, int v2) {
        // implement code
}
```

Câu 7: (1) Viết phương tính tổng bậc của mỗi đỉnh

```
public int deg(int v)  {
        // implement code
}
```

Câu 8: (1) Viết phương tính tổng bậc của đồ thị

```
public int sumDeg (int v) {
        // implement code
}
```

Câu 9: (1) Viết phương tính tổng số đỉnh của đồ thị

```
public int numVertexs() {
        // implement code
}
```

Câu 10: (1) Viết phương tính tổng số cạnh của đồ thị

```
public int numEdges(){
        // implement code
}
```

Câu 11: (1) Viết phương thức kiểm tra đồ thị có liên thông hay không

```
public boolean checkConnect(){
        // implement code
}
```

Câu 12: (2) Viết phương thức xét tính liên thông của đồ thị: số thành phần liên thông, liệt kê các đỉnh thuộc từng thành phần liên thông nếu có?

```
public void diTimCacDinhLienThong(….){
// implement code
}
public void xetTinhLienThong(….){
// implement code
}
```

Câu 13: (2) Viết phương thức dùng giải thuật BFS duyệt đồ thị G,

```
public int[]//void BFSGraph(){
// implement code
}
public int[]//void BFSGraph(int startVex) {
// implement code
}
```

Câu 14: (2) Viết phương thức dùng giải thuật DFS duyệt đồ thị G,

```
public int[]//void DFSGraph() {
// implement code
}
public int[]//void DFSGraph( int startVex) {
// implement code
}
```

Câu 15: (2) Viết phương thức kiểm tra đồ thị có liên thông hay không bằng cách sử dụng thuật toán duyệt theo chiều rộng hoặc duyệt theo chiều sâu?

```
public boolean isConnected() {
// implement code
}
```

Câu 16: (2) Viết phương thức tìm đường đi giữa 2 đỉnh từ s tới t bằng cách sử dụng thuật toán duyệt theo chiều rộng hoặc duyệt theo chiều sâu?

```
public void findPathTwoVexs(int s, int t) {
// implement code
}
```

Câu 17: (2) Viết phương thức kiểm tra đồ thị có lưỡng phân hay không bằng cách sử dụng thuật toán duyệt theo chiều rộng hoặc duyệt theo chiều sâu để tô màu cho 2 đỉnh có tạo cạnh với nhau?

```
public boolean checkBipartiteGraph() {
// implement code
}
```

Câu 18: (3) Viết phương thức kiểm tra đồ thị G có chu trình Euler hay không?

```
public boolean isEulerGraph(….){
        // implement code
}
```

Câu 19: (3) Viết phương thức kiểm tra đồ thị G có đường đi Euler hay không?

```
public boolean isHalfEulerGraph(…){
        // implement code
}
```

Câu 20: (3) Viết phương thức tìm chu trình Euler của đồ thị G?

```
public void  findEulerCycle(…) {
        // implement code
}
```

Câu 21: (3) Viết phương thức tìm đường đi Euler của đồ thị G?

```
public void findEulerPath(…) {
        // implement code
}
```

Câu 22: (4) Viết phương thức kiểm tra đồ thị G có đường đi Hamilton hay không?

```
public boolean isHalfHamiltonGraph(…){
        // implement code
}
```

Câu 23: (4) Viết phương thức kiểm tra đồ thị G có chu trình Hamilton hay không?

```
public boolean isHamiltonGraph(…){
        // implement code
}
```

Câu 24: (4) Viết phương thức tìm chu trình Hamilton của đồ thị G?

```
public void  findHamiltionCycle(…) {
        // implement code
}
```

Câu 25: (4) Viết phương thức tìm đường đi Hamilton của đồ thị G?

```
public void findHamiltionPath(….) {
        // implement code
```

```
}
```

Câu 26: (5) Viết phương thức duyệt cây bao trùm bằng thuật toán duyệt theo chiều sâu DFS đệ quy?

```
public int[][] SpanningTreeByDFS(int v){
        // implement code
}
```

Câu 27: (5) Viết phương thức duyệt cây bao trùm bằng thuật toán duyệt theo chiều sâu DFS khử đệ quy?

```
public int[][] SpanningTreeByDFS(int v){
        // implement code
}
```

Câu 28: (5) Viết phương thức duyệt cây bao trùm bằng thuật toán duyệt theo chiều rộng BFS?

```
public int[][] SpanningTreeByBFS(int[][] matrix, int v){
        // implement code
}
```

Câu 29: (6) Viết phương thức xác định Cây bao trùm có chu trình hay không?

```
public boolean checkCycle(int[][] matrix, int v1, int v2){
        // implement code
}
```

Câu 30: (6) Viết phương thức tìm cây bao trùm có trọng số nhỏ nhất bằng thuật toán Kruskal?

```
public int[][] SpanningTreeByKruskal (int[][] matrix){
        // implement code
}
```

Câu 31: (6) Viết phương thức tìm cây bao trùm có trọng số nhỏ nhất bằng thuật toán Prim?

```
public int[][] SpanningTreeByPrim (int[][] matrix, int verStart){
        // implement code
}
```

Câu 32: (7) Viết phương thức tìm đường đi ngắn nhất từ một đỉnh cho trước đến tất cả các đỉnh của đồ thị bằng thuật toán Disktra?

```
public void algoDisktra(int[][] matrix, int verStart){
        // implement code
```

```
}
```

Câu 33: (7) Viết phương thức tìm đường đi ngắn nhất từ đỉnh **src** đến **des** bằng thuật toán Disktra?

```
public abstract void findABbyDisktra(int[][] matrix, int src, int des){
    // implement code
}
```

Câu 34: (8) Viết phương thức tìm đường đi ngắn nhất giữa các cặp đỉnh của đồ thị bằng thuật toán Floyd?

```
public abstract void algoFloyd (int[][] matrix){
    // implement code
}
```

Câu 35: (8) Viết phương thức tìm đường đi ngắn nhất giữa các cặp đỉnh của đồ thị bằng thuật toán Floyd mở rộng (in ra ma trận khoảng cách và ma trận chứa dỉnh liền trước)?

```
public abstract void algoFloyd (int[][] matrix){
    // implement code
}
```

Câu 36: (8) Viết phương thức in ra đường đi giữa 2 đỉnh bất kỳ bằng thuật toán Floyd mở rộng (in ra ma trận khoảng cách và ma trận chứa dỉnh liền trước)?

```
public abstract void algoFloyd (int[][] matrix, int startV, int endV){
    // implement code
}
```

Câu 37: (9) Viết phương thức in ra đường từ một đỉnh cho trước đến tất cả các đỉnh của đồ thị bằng thuật Bellman-Frod, và in ra tất cả các đường đi?

```
public abstract void algoBellmanFord (int[][] matrix, int startV){
    // implement code
}
```

Câu 38: (9) Viết phương thức tìm ma trận khả liên của đồ thị bằng thuật toán tính bao đóng bắt cầu- WarShall, và in ra tất cả các đường đi?

```
public abstract void algoWarShall(int[][] matrix, int startV){
    // implement code
}
```