```
import java.io.*;
```

- **Đọc/ghi file text:**

```
String s;

String[] words;

 try (BufferedReader br = new BufferedReader(new FileReader(path))) { String line =br.readLine();

   while ((s = br.readLine()) !=null){

   words = s.split("\t"); }} catch (IOException exc) {System.out.println("I/O Error: " + exc);}

try (FileWriter fout = new FileWriter("./src/DanhsachHS.txt")) {

fout.write(s);
```

- **Đọc/ghi binary:**

```
try (ObjectInputStream fin = new ObjectInputStream(new FileInputStream(file_name))) {...} catch (...) {...}

boolean is_eof = false;

while (!is_eof) {

       Object obj = fin.readObject();

       if (obj != null) T ten_bien = (T) obj; // Ép kiểu

       else is_eof = true;

}

try (ObjectOutputStream fout = new ObjectOutputStream(new FileOutputStream(file_name))) {...} catch (...) {...}

T ten_bien = new T;
fout.writeObject(ten_bien);

fout.close();
```

- **Generic**: <**?** extends **superclass**> : chấp nhận class kế thừa class cha

  <**?** super **subclass**> : chấp nhận class được kế thừa từ class con

```
import java.util.*;

public static <E extends Comparable<E>> E getMax(E[] list) {

 E max = list[0];

 for (int i = 1; i < list.length; i++){

    ([i].compareTo(max) > 0)

        max = list[i];}

        return max;}

static class A implements Comparable<A> {

       @overide

       public int compareTo(T obj_1) {} // return (1 : this > obj_1, 0: bằng , -1: nhỏ hơn)});

@Override

  public String toString() {}
```

- **JDBC:**

```
stmt = conn.createStatement();

String sql = "";//sql = "INSERT INTO COLUMN VALUES (?,?,?,?,?);";

stml.setString(1, (String) row[0]);//

stmt.executeUpdate(sql);

Update: update (tên table) set … where

Delete : delete from (tên table) where

Insert : insert into (tên table)  (giá trị 1, giá trị 2,...)

import java.sql.*;

static Connection getConnection(){

Connection conn = null;

try{

String JDBC_DRIVER="com.microsoft.sqlserver.jdbc.SQLServerDriver";

 String DB_URL ="jdbc:sqlserver://localhost:1433”+ "databaseName=QLT”+"integratedSecurity=true";

  // STEP 2: Register JDBC driver

Class.forName(JDBC_DRIVER);

// STEP 3: Open a connection

 conn = DriverManager.getConnection(DB_URL, USER_NAME, PASSWORD);}catch (SQLException | ClassNotFoundException exc) {System.err.println("Got an exception!, Cannot connect to server! ");}

return conn;}

get result from query

Connection conn =Connections.getConnection();

PreparedStatement stm = null;

ResultSet rs = null;

String query = "SELECT * FROM THI_SINH";

try {

stm = conn.prepareStatement(query);

rs = stm.executeQuery();

while (rs.next()) {

Object[] row = new Object[n];

       row[0] = rs.getString("ID");rs.getInt("ID");
```

- **Swing:**

```
import java.awt.*;

import java.swing.*;

public class GUI  {

       GUI() {

JFrame frame = new JFrame("Simple frame");frame.setLayout(null);

--Khởi tạo các component

--setBounds

-- add vào frame, panel

frame.setSize(900, 400);

frame.setLocationRelativeTo(null);

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

frame.setVisible(true);

-- Thêm sự kiện

button.addActionListener(new ActionListener() {

String command = this.getActionCommand();

public void actionPerformed(ActionEvent evt) {// do something}});

// Menu, Thêm sự kiện như button

JMenuBar jmb = new JMenuBar();

// Create the File menu và Set phím tắt

JMenuBar mb=new JMenuBar();

menu=new JMenu("Menu");

submenu=new JMenu("Sub Menu");

i1=new JMenuItem("Item 1");

menu.add(i1);
submenu.add(i4); submenu.add(i5);

menu.add(submenu);
```

```
mb.add(menu);

f.setJMenuBar(mb);
```

- **Thread:**

```
class A{

synchronized int method(int nums[])
{}}

class MyThread implements Runnable {

Thread thrd;

static A sa = new A();

MyThread4(String name, int nums[]) {

thrd = new Thread(this, name);

a = nums; thrd.start(); // start the
thread}

public void run() {}

}

public static void main(String args[])
{

MyThread4 mt1 = new MyThread4("Child
#1");

try {

mt1.thrd.join();} catch
(InterruptedException exc) {

System.out.println("Main thread
interrupted.");}}
```

- **Network:**

```
import java.io.*;

import java.net.*;
```

TCP

Server

```
    public static void main(String[]
args) throws IOException

    { ServerSocket ss = new
ServerSocket( số port);

   Socket s; s = ss.accept();

   DataInputStream dis = new
DataInputStream(s.getInputStream());

   DataOutputStream dos = new
DataOutputStream(s.getOutputStream());

        String mess=dis.readUTF();
//nhận tin nhắn

        dos.writeUTF(...);  // gửi tin
nhắn

    }
```

Client

```
    public static void main(String
args[]) throws UnknownHostException,
IOException

    {

        InetAddress ip =
InetAddress.getByName("localhost");

        Socket s = new Socket(ip, (số
port) );

        DataInputStream dis = new
DataInputStream(s.getInputStream());

        DataOutputStream dos = new
DataOutputStream(s.getOutputStream());

        String mess=dis.readUTF();

        dos.writeUTF(...);

}
```

UDP:

Server:

```
public static void main(String args[])
throws Exception {

    DatagramSocket serverSocket = new
DatagramSocket(số port);

    byte[] receiveData = new
byte[1024];

    byte[] sendData  = new byte[2048];

    DatagramPacket receivePacket = new
DatagramPacket(receiveData,
receiveData.length);


serverSocket.receive(receivePacket);
//nhận gói tin qua phương thức
receive()

    String sentence = new
String(receivePacket.getData());//Chuy
ển dữ liệu nhận về dạng String

    InetAddress IPAddress =
receivePacket.getAddress(); //Lấy địa
chỉ IP của bên gửi

    int port =
receivePacket.getPort();  //Lấy số
hiệu cổng bên gửi

    sendData = sentence.getBytes();
//tạo gói tin để gửi đi client

    DatagramPacket sendPacket = new
DatagramPacket(sendData,
sendData.length, IPAddress, port);

    serverSocket.send(sendPacket);

    serverSocket.close();   //Gửi gói
tin đi

   }
```

Client:

```
public static void main(String args[])
throws Exception {

   DatagramSocket clientSocket = new
DatagramSocket( số port);

   InetAddress IPAddress =
InetAddress.getByName("localhost");

      //Tạo dữ liệu(group of bytes)
cho gói tin nhận và gói tin gửi

      byte[] sendData = new
byte[1024];

      byte[] receiveData = new
byte[2048];

      //Lấy dòng văn bản nhập từ bàn
phím và gán cho biến sentence

      BufferedReader inFromUser = new
BufferedReader(new
InputStreamReader(System.in));

      String sentence =
inFromUser.readLine();

      sendData = sentence.getBytes();
//chuyển thành mảng byte

      DatagramPacket sendPacket = new
DatagramPacket(sendData,sendData.lengt
h, IPAddress, 9876);

      clientSocket.send(sendPacket);

      DatagramPacket receivePacket
=new DatagramPacket(receiveData,
receiveData.length);

      //Lấy biến receivePacket để nhận
gói tin bằng phương thức receive()


clientSocket.receive(receivePacket);

      String modified_Sentence = new
String(receivePacket.getData());

      clientSocket.close();

    }
```