

## HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG KHOA CÔNG NGHỆ THÔNG TIN

# BÁO CÁO THỰC TẬP CƠ SỞ

## Đề tài: Python và xử lí ảnh

**Giảng viên hướng dẫn : TS. Nguyễn Tất Thắng Sinh viên thực hiện : Đào Trọng Phúc**

**Mã Sinh viên : B19DCCN504**

**Lớp : D19CNPM8**

**Hệ Đại học : Chính quy**

**Niên khoá: : 2019-2024**

**Hà Nội – Tháng 6, năm 2022**

# MỤC LỤC

1. [PHẦN MỞ ĐẦU 2](#_bookmark0)
   1. [Giới thiệu về ngôn ngữ lập trình Java 2](#_bookmark1)
   2. [Mục đích 2](#_bookmark2)
   3. [Phạm vi tìm hiểu 2](#_bookmark3)
   4. [Phương pháp 2](#_bookmark4)
2. [PHẦN NỘI DUNG 3](#_bookmark5)
   1. [Ôn và nắm lại kiến thức 3](#_bookmark6)
      1. [Kĩ năng lập trình 3](#_bookmark7)
      2. [Exception trong Java 3](#_bookmark8)
      3. [Các class căn bản trong Java 3](#_bookmark9)
      4. [JavaSwing 3](#_bookmark10)
      5. [Input/Output trong Java 4](#_bookmark11)
      6. [Java bean 4](#_bookmark12)
   2. [Bài tập 4](#_bookmark13)
      1. [Bài tập kỹ năng lập trình 4](#_bookmark14)
      2. [Bài tập Exception 8](#_bookmark15)
      3. [Bài tập về các class căn bản trong Java 9](#_bookmark16)
      4. [Bài tập về Java swing 12](#_bookmark17)
      5. [Input/Output 14](#_bookmark18)
      6. [Java bean 19](#_bookmark19)
   3. [Project cuối khóa 20](#_bookmark20)
   4. [Tổng kết 27](#_bookmark21)

# PHẦN MỞ ĐẦU

## Giới thiệu về ngôn ngữ lập trình Python

### **Python** là một ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình và là ngôn ngữ lập trình dễ học; được dùng rộng rãi trong phát triển trí tuệ nhân tạo. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu.

### **Python** hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl. Python được phát triển trong một dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

## Giới thiệu các module sử dung :

## Phạm vi tìm hiểu

* + Java core

## Phương pháp

* + Tìm hiểu lý thuyết kết hợp bài tập vận dụng
  + Tự tạo sản phẩm cuối khóa sử dụng Java core

# PHẦN NỘI DUNG

## Ôn và nắm lại kiến thức

### Kĩ năng lập trình

* + - Tìm hiểu môi trường làm việc Java: các khái niệm JDK, JVM, Java API, trình Java compiler (javac), biên diichj compiler và thực thi lớp java từ dòng lệnh.
    - Các kiểu dữ liệu cơ bản trong java: int, long, char, double
    - Mô hình sử dụng cho các biến số (static, khai báo biến toàn cục, cục bộ)
    - Các toán tử trong java (toán tử logic, toán tử điều kiện)
    - Các cấu trúc điều khiển cơ bản trong java, foreach (java 5 trở lên)
    - Các hàm toán học cơ bản class Math: max, min, ceil, floor, round, pow, random
    - Các khái niệm cơ bản: class, kế thừa (kế thừa), đồ vật (trừu tượng), đa hình (đa hình), interface, constructor, từ khóa cuối cùng. Ý nghĩa của chúng

trong công việc hóa các đối tượng.

* + - Khái niệm overload, override.
    - Ép kiểu (casting), từ khóa instance of, auto-boxing (Java 5 trở lên)
    - Khái niệm về lớp phạm vi (public, private, lồng nhau lớp), phạm vi method (public, private, protected).
    - Khái niệm, cách sử dụng GenericType (Java 5 trở lên)

### Exception trong Java

* + - Khái niệm , mục đích sử dụng.
    - Các kiểu Exception trong Java.
    - Điều khiển các Exception ( các từ khóa try , catch , throw , throws , finally)
    - Multiple catch block
    - Tự định nghĩa một Exception và sử dụng với khai báo throw vàthrows
    - Tìm hiểu 1 số exception đơn giản cơ bản

### Các class căn bản trong Java

* + - Array(khởi tạo, truy xuất, độ dài, mang nhiều chiều)
    - Collection, Set, List, Map
    - Hash Set, ArrayList, HashTable(for, foreach, iterator, enumaration)
    - Integer, Long, Double,…
    - String, StringBuffer, StringBuilder, StringTokenizer, Date, Calendar, SimpleDateFormat, RegularExpression

### JavaSwing

* + - Các khung chứa: Jframe, Jpanel, JDialog
    - Các component đơn giản: Jbutton, Jlable, Jtextfield, Jcombobox, Jlist
    - Xử lý sự kiện các component
    - Các layout, cách sử dụng

### Input/Output trong Java

* + - Khái niệm về stream
    - OutputStream(ByteArrayOutputStream, FileOutputStream, BufferedOutputStream, DataOutputStream)
    - InputStream (ByteArrayInputStream, FileInputStream, BufferedInputStream , DataOutputStream )
    - File, Folder
    - Writer, Reader

### Java bean

* + - Khái niệm và cách sử dụng
    - Setter, Getter

## Bài tập

### Bài tập kỹ năng lập trình

* Viết chương trình mô phỏng giải thuật Binary search

Class BinarySearch để thực hiện mô phỏng lại giải thuật tìm kiếm nhị phân

/\*\*

* Copyright(C) 2021 Luvina Software Company
* BinarySearch.java, Jul 13, 2021 Đức Nguyễn

\*/

**package** bai1;

/\*\*

* **@author** DucNT
* Tìm kiếm nhị phân

\*/

**public class** BinarySearch {

/\*\*

* + Tìm vị trí xuất hiện phần tử cần tìm trong mảng
  + **@param** array: mang cac so nguyen
  + **@param** x : gia tri can tim
  + **@param** low : phan tu dau tien cua mang
  + **@param** high : phan tu cuoi cung cua mang
  + **@return** result : vi tri phan tu can tim

\*/

**int** binarySearch(**int** array[], **int** x, **int** low, **int** high) {

**int** result = -1;

// kiểm tra phần tử đầu cuối cùng lớn hơn phần tử đầu tiên

**if** (high >= low) {

**int** mid = low + (high - low) / 2;

// Kiểm tra phần tử ở giữa bằng giá trị cần tìm

**if** (array[mid] == x) { result = mid;

}

// Kiểm tra phần tử ở giữa lớn hơn giá trị cần tìm

**else if** (array[mid] > x) {

result = binarySearch(array, x, low, mid - 1);

}

#### else {

result = binarySearch(array, x, mid + 1, high);

}

}

**return** result;

}

}

### Class Main để thực thi chương trình

/\*\*

* Copyright(C) 2021 Luvina Software Company
* Main.java, Jul 13, 2021 Đức Nguyễn

\*/

**package** bai1;

/\*\*

* **@author** DucNT
* Chạy chương trình

\*/

**public class** Main {

/\*\*

* + Chạy chương trình với dữ liệu đã cho
  + **@param** args

\*/

**public static void** main(String args[]) {

// Khai báo đối tượng BinarySearch BinarySearch ob = **new** BinarySearch();

// Khởi tạo mảng array

**int** array[] = { 3, 4, 5, 6, 7, 8, 9 };

// Khởi tạo độ dài của mảng

**int** n = array.length;

// Khởi tạo giá trị phần tử cần tìm

**int** x = 4;

// Tìm vị trí của phần tử

**int** result = ob.binarySearch(array, x, 0, n - 1);

**if** (result == -1) System.***out***.println("Not found");

#### else

System.***out***.println("Element found at index " + result);

}

### Ôn tập về các đặc trưng trong ngôn ngữ Java Abstract class Vehicle

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + Vehicle.java, Jul 13, 2021 Đức Nguyễn

\*/

**package** bai2;

/\*\*

* + **@author** DucNT
  + Lớp trừu tượng Vehicle

\*/

**public abstract class** Vehicle {

/\*\*

\*

* + - **@param** batDau
    - **@param** ketThuc

\*/

**public abstract void** move(**int** batDau, **int** ketThuc);

/\*\*

\*

* + - **@param** chieuDai
    - **@param** chieuRong

\*/

**public abstract void** setSize(**int** chieuDai, **int** chieuRong);

/\*\*

\*

* + - **@param** mau

\*/

**public abstract void** setColor(String mau);

}

### Class Boat kế thừa từ lớp cha Vehicle

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + Boat.java, Jul 13, 2021 Đức Nguyễn

\*/

**package** bai2;

/\*\*

* + **@author** DucNT
  + Khai báo lớp Boat kế thừa từ lớp Vehicle

\*/

**public class** Boat **extends** Vehicle{

/\*\*

* + - **@param** batDau
    - **@param** ketThuc

\*/ @Override

**public void** move(**int** batDau, **int** ketThuc) { System.out.println("Method [move] from class [Boat]");

}

/\*\*

* + - **@param** chieuDai
    - **@param** chieuRong

\*/ @Override

**public void** setSize(**int** chieuDai, **int** chieuRong) { System.out.println("Method [setSize] from class [Boat]");

}

/\*\*

* + - **@param** mau

\*/ @Override

**public void** setColor(String mau) {

System.out.println("Method [setColor] from class [Boat]");

}

}

### Lớp con Car kế thừa từ lớp cha Vehicle

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + Car.java, Jul 13, 2021 Đức Nguyễn

\*/

**package** bai2;

/\*\*

* + **@author** DucNT
  + Khai báo lớp Car kế thừa từ lớp Vehicle

\*/

**public class** Car **extends** Vehicle{

/\*\*

* + - **@param** batDau
    - **@param** ketThuc

\*/ @Override

**public void** move(**int** batDau, **int** ketThuc) {

}

/\*\*

* + - **@param** chieuDai
    - **@param** chieuRong

\*/ @Override

**public void** setSize(**int** chieuDai, **int** chieuRong) { System.***out***.println("Method [setSize] from class [Car]");

}

/\*\*

* + - **@param** mau

\*/ @Override

**public void** setColor(String mau) {

System.***out***.println("Method [setColor] from class [Car]");

}

}

### Lớp con Ferrary kế thừa từ lớp cha Vehicle

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + Ferrary.java, Jul 13, 2021 Đức Nguyễn

\*/

**package** bai2;

/\*\*

* + **@author** DucNT
  + Khai báo lớp Ferrary kế thừa từ lớp Vehicle

\*/

**public class** Ferrary **extends** Car{

/\*\*

* + - **@param** batDau
    - **@param** ketThuc

\*/ @Override

**public void** move(**int** batDau, **int** ketThuc) {

**super**.move(batDau, ketThuc);

System.***out***.println("Method [move] from class [Ferrary]");

}

/\*\*

* + - **@param** chieuDai
    - **@param** chieuRong

\*/ @Override

**public void** setSize(**int** chieuDai, **int** chieuRong) { **super**.setSize(chieuDai, chieuRong); System.***out***.println("Method [setSize] from class [Ferrary]");

}

/\*\*

\*

* + - * **@param** diemDau
      * **@param** diemCuoi

\*/

**public void** moveRelative(**int** diemDau, **int** diemCuoi){ System.***out***.println("Method [moveRelative] from class [Ferrary]");

}

}

### Bài tập Exception

* Tự định nghĩa Exception

Class ExceptionDemo tự định nghĩa ra exception

/\*\*

* Copyright(C) 2021 Luvina Software Company
* TestCustomException1.java, Jul 14, 2021 Đức Nguyễn

\*/

**package** bt2;

/\*\*

* **@author** DucNT
* Chương trình chính, các ngoại lệ sẽ được xử lý bởi MyException

\*/

**public class** ExceptionDemo {

/\*\*

\*

* + **@param** age: tuổi nhập vào
  + **@throws** MyException

\*/

**public static void** validate(**int** age)**throws** MyException{ MyException myException = **new** MyException();

/\*\*

\* Kiểm tra số tuổi có hợp lệ hay không

\*/

**if**(age<18) {

myException.showException("Age invalid");

}

#### else {

System.***out***.println("Age valid");

}

}

}

### Class MyException kế thừa từ lớp Exception để xử lý khi có ngoại lệ xảy ra

/\*\*

* Copyright(C) 2021 Luvina Software Company
* InvalidAgeException.java, Jul 14, 2021 Đức Nguyễn

\*/

**package** bt2;

/\*\*

* **@author** DucNT
* Định nghĩa class MyException xử lý các ngoại lệ

\*/

**public class** MyException **extends** Exception{ MyException(){

}

/\*\*

* + xử lý ngoại lệ
  + **@param** message : in ra message khi ngoại lệ được xử lý

\*/

**public void** showException(String message) { System.***out***.println(message);

}

}

### Class Main thực thi chương trình

/\*\*

* Copyright(C) 2021 Luvina Software Company
* Main.java, Jul 14, 2021 Đức Nguyễn

\*/

**package** bt2;

/\*\*

* **@author** DucNT
* Chạy chương trình

\*/

**public class** Main {

/\*\*

* + Đưa giá trị cần kiểm tra vào và chạy chương trình
  + **@param** args

\*/

**public static void** main(String args[]){

**try**{

ExceptionDemo.*validate*(13);

}**catch**(Exception m){System.***out***.println("Exception occured: "+m);}

System.***out***.println("rest of the code...");

}

}

### Bài tập về các class căn bản trong Java

* Trò chơi đoán số

Class GuessingGame để mô phỏng quá trình dự đoán số bằng cách sinh ra số ngẫu nhiên

/\*\*

* Copyright(C) 2021 Luvina Software Company
* GuessingGame.java, Jul 16, 2021 Đức Nguyễn

\*/

**package** bai1;

**import** java.util.Random;

**import** java.util.Vector;

/\*\*

* **@author** Dell

\*

\*/

**public class** GuessingGame {

**private static int** *target*;

**static** Vector<Integer> *guesses* ;

**public** GuessingGame() {

}

/\*\*

* + constructor cho class GuessingGame
  + **@param** target : giá trị được tìm kiếm
  + **@param** guesses : vector các giá trị được đưa vào để so sánh

\*/

**public** GuessingGame(**int** target, Vector<Integer> guesses) {

**this**.*target* = target;

**this**.*guesses* = **new** Vector<>(100, 25);

}

/\*\*

* + sinh ra các số ngẫu nhiên trong khoảng 1 đến 100
  + **@param** target : giá trị được tìm kiếm

\*/

**public static void** startGuessing(**int** target){

/\*\*

\* in ra số ngẫu nhiên đầu tiên

\*/

Random rand = **new** Random();

**int** ranNum = rand.nextInt(100)+1;

/\*\*

* + - Nếu số sinh ra nằm trong khoảng 1 đến 100 và khác với giá trị cần tìm kiếm

\*/

**while** (1< ranNum && ranNum<100 && ranNum != target){

/\*\*

\* tiếp tục sinh ra số tiếp theo

\*/

ranNum = rand.nextInt(100)+1;

/\*\*

\* thêm số vừa sinh ra vào vector

\*/

*guesses*.add(ranNum);

}

*guesses*.add(target);

}

/\*\*

* + in ra các giá trị trong vector

\*/

**public static void** printGuess(){

**for** (**int** i=0;i< *guesses*.size();i++){ System.***out***.print(*guesses*.get(i) + " ");

}

/\*\*

* + - in ra số phần tử trong vector

\*/

System.***out***.println("size of vector: "+ (*guesses*.size()-1));

}

/\*\*

* + chạy chương trình
  + **@param** args

\*/

}

### Class Main để thực thi chương trình

/\*\*

* Copyright(C) 2021 Luvina Software Company
* Main.java, Jul 20, 2021 Đức Nguyễn

\*/

**package** bai1;

**import** java.util.Scanner;

/\*\*

* **@author** Dell

\*

\*/

**public class** Main {

**public static void** main(String[] args) { String s;

#### try {

Scanner scanner = **new** Scanner(System.***in***); s = scanner.next();

**int** target = Integer.*parseInt*(s);

GuessingGame guessingGame = **new** GuessingGame(target, GuessingGame.*guesses*);

guessingGame.*startGuessing*(target); guessingGame.*printGuess*();

} **catch** (NumberFormatException e) { System.***out***.println(e);

}

}

}

### Bài tập về biểu thức chính quy(regular expression)

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + Main.java, Jul 16, 2021 Đức Nguyễn

\*/

/\*\*

* + **@author** Dell

\*

\*/

**public class** Main {

**public static void** main(String[] args) {

/\*\*

\* định dạng số điện thoại với mỗi phần tử có ít nhất 4 chữ số

\*/

String regex1 = "[0-9]{4,}-[0-9]{4,}-[0-9]{4,}";

String s1 = "0123-4567-8901";

**if** (s1.matches(regex1)) {

System.***out***.println("match");

}

#### else {

System.***out***.println("not match");

}

/\*\*

* Định dạng [XXXX@YY.ZZ](mailto:XXXX@YY.ZZ)
* XXXX có ít nhất 4 kí tự và ko chứa ;@.,\*
* YY có ít nhất 2 kí tự chỉ chứa chữ và số
* ZZ có ít nhất 2 kí tự và chỉ chứa chữ

\*/ String regex =

"([[\\w][^;@.,\*][^^]]{4,})([@])([\\w]{2,})([.])([/\b[^\\d\\W]+\b/g]{2,}

)";

String s = "0^ [\_ad@g2mail.com2](mailto:_ad@g2mail.com2)";

**if** (s.matches(regex)){ System.***out***.println("match");

}

#### else

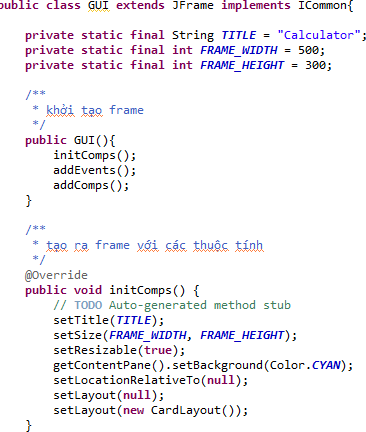
System.***out***.println("not match");

}

}

### Bài tập về Java swing

* Ứng dụng máy tính

Class GUI tạo khung cho ứng dụng

Interface Icommon định nghĩa ra các phương thức để xử lý cho ứng dụng

/\*\*

* Copyright(C) 2021 Luvina Software Company
* ICommon.java, Jul 29, 2021 Đức Nguyễn

\*/

**package** view;

/\*\*

* **@author** Dell
* tạo ra các phương thức để xử lý logic

\*/

**public interface** ICommon {

**void** initComps(); **void** addEvents(); **void** addComps();

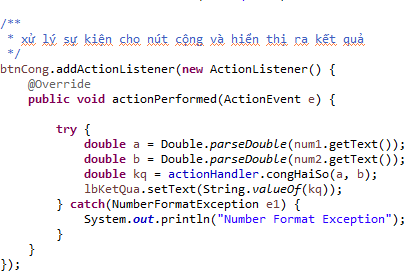
}

### Class MainPanel định nghĩa ra giao diện chính của ứng dụng chứa các component liên quan

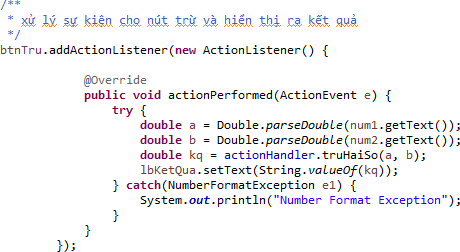
Tạo ra panel với các component



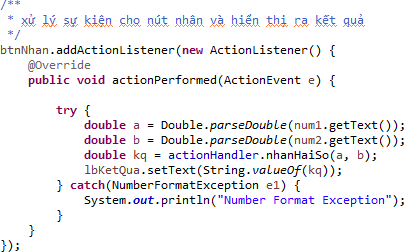
Xử lý sự kiện cho nút cộng

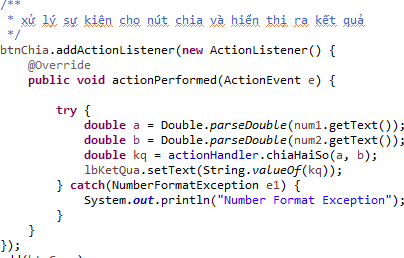


Xử lý sự kiện cho nút trừ



Xử lý sự kiện cho nút nhân



Xử lý sự kiện cho nút chia

### Input/Output

* Đọc nội dung từ file

Class FileInput để đọc nội dung từ file

/\*\*

* Copyright(C) 2021 Luvina Software Company
* FileInput.java, Jul 22, 2021 Đức Nguyễn

\*/

**package** bai1;

**import** java.io.File;

**import** java.io.FileInputStream; **import** java.io.FileNotFoundException; **import** java.util.Scanner;

/\*\*

* **@author** Dell
* thực hiện đọc vào file và thực hiện các thao tác

\*/

**public class** FileInput {

**public static** File inputFile = **new** File("C:\\Users\\Dell\\eclipse- workspace\\DucNT\_Input\_Out\\src\\bai1\\input.txt");

**public static** FileInputStream in = **null**; **public static** String line[] = **new** String[1000]; **static int** i = 0;

**static int** dem = 0;

**static int** kt = 0;

**public static void** show() {

#### try {

/\*\*

\* tạo đối tượng đọc file

\*/

in = **new** FileInputStream(inputFile); Scanner scanner = **new** Scanner(in, "UTF-8");

/\*\*

\* đọc hết 1 dòng

\*/

**while** (scanner.hasNextLine()) { line[i] = scanner.nextLine(); **if** (line[i].trim() != "") {

/\*\*

\* tách dòng ra thành mảng các từ

\*/

String item[] = line[i].split(" ");

**for** (**int** j = 0;j<item.length;j++) kt+=item[j].length();

dem += item.length;

} i++;

}

System.out.println("So luong dong: " + i); System.out.println("So luong ki tu: " + kt); System.out.println("So luong tu: " + dem);

} **catch** (FileNotFoundException e) { System.out.print(e);

}

}

}

### Class Main để thực thi chương trình

/\*\*

* Copyright(C) 2021 Luvina Software Company
* Main.java, Jul 22, 2021 Đức Nguyễn

\*/

**package** bai1;

/\*\*

* **@author** Dell

\*

\*/

**public class** Main {

/\*\*

* + chạy chương trình
  + **@param** args

\*/

**public static void** main(String[] args) { FileInput fileInput = **new** FileInput(); fileInput.*show*();

}

}

### Đọc ghi đối tượng bất kỳ Class User

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + User.java, Jul 22, 2021 Đức Nguyễn

\*/

**package** bai3;

**import** java.io.Serializable;

/\*\*

* + **@author** Dell

\*

\*/

**public class** User **implements** Serializable{

**private** String username;

**private int** age; **private** String address; **private float** score;

**public** User() {

}

**public** User(String username, **int** age, String address, **float** score)

{

**this**.username = username; **this**.age = age; **this**.address = address; **this**.score = score;

}

**public** String getUsername() {

**return** username;

}

**public void** setUsername(String username) {

**this**.username = username;

}

**public int** getAge() {

**return** age;

}

**public void** setAge(**int** age) {

**this**.age = age;

}

**public** String getAddress() {

**return** address;

}

**public void** setAddress(String address) {

**this**.address = address;

}

**public float** getScore() {

**return** score;

}

**public void** setScore(**float** score) {

**this**.score = score;

}

@Override

**public** String toString() {

**return** "User{" +

"username='" + username + '\'' + ", age=" + age +

", address='" + address + '\'' + ", score=" + score +

'}';

}

}

### Class FileIO để xử lý file

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + FileIO.java, Jul 22, 2021 Đức Nguyễn

\*/

**package** bai3;

**import** java.io.FileInputStream; **import** java.io.FileNotFoundException; **import** java.io.FileOutputStream; **import** java.io.IOException;

**import** java.io.ObjectInputStream; **import** java.io.ObjectOutputStream; **import** java.util.ArrayList; **import** java.util.Collections; **import** java.util.List;

/\*\*

* + **@author** Dell
  + đọc ghi file và xử lý các yêu cầu

\*/

**public class** FileIO {

/\*\*

\* tạo ra list các đối tượng User

\*/

**public** List<User> users = **new** ArrayList<>(); User user = **new** User("nam", 20, "adaa", 1); User user1 = **new** User("man", 32, "dada", 2);

/\*\*

\* mở luồng ghi danh sách User vào file

\*/

**public void** writeFile(){ users.add(user); users.add(user1); **try** {

FileOutputStream fos = **new**

FileOutputStream("C:\\Users\\Dell\\IdeaProjects\\BTLuvina\\src\\bt22\_7\

\bt3\\input.txt");

ObjectOutputStream oos = **new** ObjectOutputStream(fos); oos.writeObject(users);

} **catch** (FileNotFoundException e){ System.out.println(e);

} **catch** (IOException e) { e.printStackTrace();

}

}

/\*\*

\* mở luồng đọc từ file ra và hiển thị lên màn hình

\*/

**public void** readFile(){

#### try {

FileInputStream fis = **new** FileInputStream("C:\\Users\\Dell\\IdeaProjects\\BTLuvina\\src\\bt22\_7\\ bt3\\input.txt");

ObjectInputStream ois = **new** ObjectInputStream(fis); List<User> userList = **new** ArrayList<>();

userList = (List<User>) ois.readObject();

/\*\*

\* sắp xếp tên theo bảng chữ cái

\*/

Collections.sort(userList,(o1, o2) -> o1.getUsername().compareTo(o2.getUsername()));

System.out.println(userList);

**float** tong = 0;

/\*\*

\* tính tổng scores

\*/

**for** (User user: userList

) {

tong+=user.getScore();

}

System.out.println("Tong cac score: " + tong);

} **catch** (FileNotFoundException e){ e.printStackTrace();

} **catch** (IOException e) { e.printStackTrace();

} **catch** (ClassNotFoundException e) { e.printStackTrace();

}

}

}

### Class Main để thực thi chương trình

/\*\*

* + Copyright(C) 2021 Luvina Software Company
  + Main.java, Jul 22, 2021 Đức Nguyễn

\*/

**package** bai3;

/\*\*

* + **@author** Dell

\*

\*/

**public class** Main {

/\*\*

* + - chạy chương trình
    - **@param** args

\*/

**public static void** main(String[] args) { FileIO fileIO = **new** FileIO();

fileIO.writeFile(); fileIO.readFile();

}

}

### Java bean

* Tạo 1 class Java bean thuần

Class Customer là 1 bean

**package** bai1;

**import** java.io.Serializable;

/\*\*

* Copyright(C) 2021 Luvina Software Company
* Customer.java, Jul 23, 2021 Đức Nguyễn

\*/

/\*\*

* **@author** Dell

\*

\*/

**public class** Customer **implements** Serializable{

#### private long iD;

**private** String name;

**public** Customer() {

}

**public** Customer(**long** iD, String name) {

**this**.iD = iD;

**this**.name = name;

}

**public long** getiD() {

**return** iD;

}

**public void** setiD(**long** iD) {

**this**.iD = iD;

}

**public** String getName() {

**return** name;

}

**public void** setName(String name) {

**this**.name = name;

}

}

### Class CustomerObserver thực hiện các phương thức liên quan đến bean đã tạo

/\*\*

* Copyright(C) 2021 Luvina Software Company
* CustomerObserver.java, Jul 23, 2021 Đức Nguyễn

\*/

**package** bai1;

/\*\*

* **@author** Dell

\*

\*/

**public class** CustomerObserver {

**public static void** printCustomer(){ Customer customer = **new** Customer(); customer.setiD(1); customer.setName("Luvina"); System.***out***.println(customer);

}

}

### Class Main để thực thi chương trình

/\*\*

* Copyright(C) 2021 Luvina Software Company
* Main.java, Jul 23, 2021 Đức Nguyễn

\*/

**package** bai1;

/\*\*

* **@author** Dell

\*

\*/

**public class** Main {

**public static void** main(String[] args) {

CustomerObserver customerObserver = **new** CustomerObserver(); customerObserver.*printCustomer*();

}

}

## Project cuối khóa

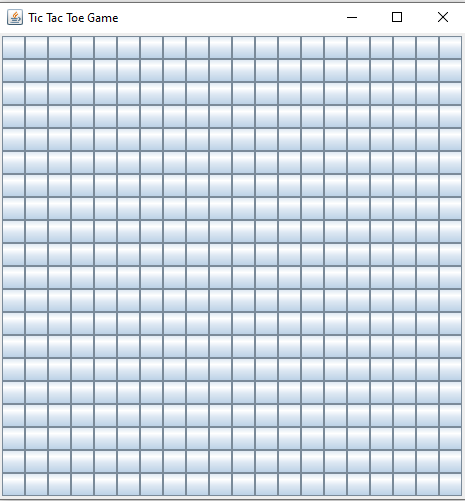
### Trò chơi cờ caro ( bàn cờ 20x20)

* Cách làm

Sử dụng Java swing để thiết kế giao diện mô phỏng bàn cờ caro Mỗi lượt chỉ có 1 người đánh 1 nước

Bên nào có được 5 nước cờ liên tục thì được tính là thắng cuộc

* Mô phỏng bàn cờ caro



* File thế cờ lưu tất cả các trường hợp có thể xảy ra trong bàn cờ
* Máy đánh nước tiếp theo ở biên để thắng theo hàng ngang, dọc và chéo sau khi đã đánh được 4 nước liên tục
* Máy đánh nước tiếp theo để chặn không cho người thắng sau khi người đã đánh được 4 nước liên tục

GGGGO GGGGO GGGGO GGGGO GGGGD

### Máy đánh nước tiếp theo vào giữa để hoàn thành 5 nước liền nhau

GGXGG GGXGG GGXGG GGDGG GGXGG

### Máy đánh nước tiếp theo vào giữa để không cho người ăn

OGGGG GOGGG GGDGG GGGOG GGGGO

### Máy đánh nước tiếp theo để ăn nước đôi

GGXGG GGXGG

GGXGG XXXDX GGXGG

### Máy đánh nước tiếp theo để chặn không cho người ăn nước đôi

GGOGG GGOGG GGOGG GGOGG OODOO

### Máy đánh nước tiếp theo để tạo thành thế 4 nước liên tiếp

GGGGG GXGGG GXGGG GXGGG GDGGG

### Máy đánh nước tiếp theo để không cho người tạo thành 4 nước liên tiếp

GGGGG GGGGG GGGGG GGGGG OODOG

### Máy đánh nước tiếp theo để tạo thành thế 3 nước liên tiếp

GGGGG GGGGG GGGGG GGGGG XXDGG

### Máy đánh nước tiếp theo để không cho người tạo thành 3 nước liên tiếp

GGGGG GGGGG GGGGG GGGGG ODOGG

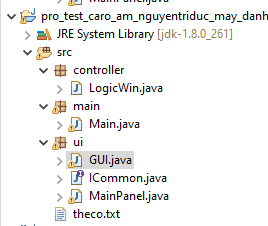
### Máy đánh nước tiếp theo để tạo thành thế 2 nước liên tiếp

GGGGG GGGDX GGGGG GGGGG GGGGG

### Máy đánh nước tiếp theo để không cho người tạo thành 2 nước liên tiếp

GGGDO GGGGG GGGGG GGGGG GGGGG

### Cấu trúc project :



Interface Icommon tạo ra các phương thức: khởi tạo view, thêm sự kiện cho view, thêm các component trên giao diện giúp cho các class view chỉ cần ghi đè lại các phương thức mà nó đã kế thừa từ interface ICommon. Việc tạo ra interface giúp cho việc nâng cấp, bảo trì, sửa đổi project được thuận tiện mà không gây ảnh

hưởng tới các thành phần khác

/\*\*

* Copyright(C) 2021 Luvina Software Company
* ICommon.java, Aug 3, 2021 Đức Nguyễn

\*/

**package** ui;

/\*\*

* **@author** Dell
* interface tạo ra các phương thức khởi tạo cho component

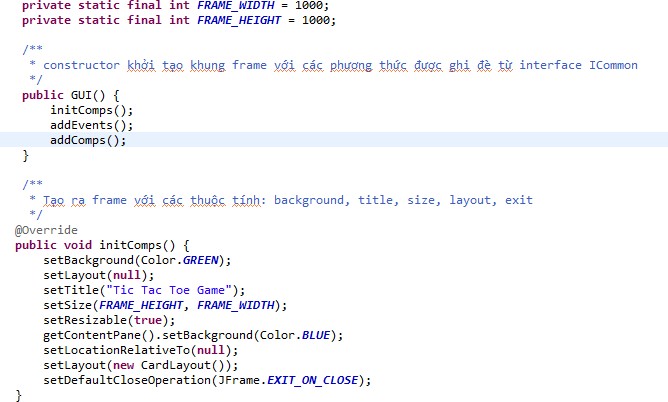
\*/

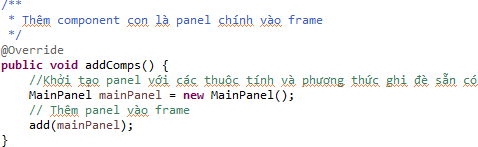
**public interface** ICommon {

**void** initComps(); **void** addEvents(); **void** addComps();

}

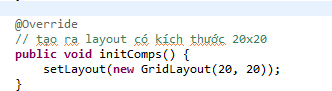
### Class GUI dùng để tạo ra khung cho giao diện của ứng dụng, các component con khi cần thêm mới chỉ cần khởi tạo và thêm mới tại class này



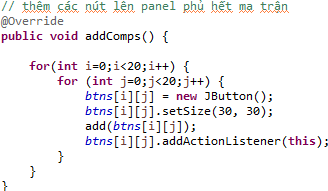


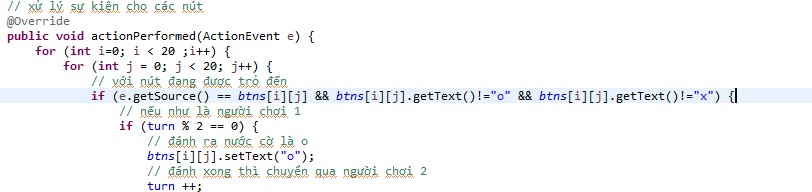
Class MainPanel hiển thị ra giao diện chính của ứng dụng

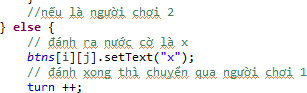
Giao diện được hiển thị dưới dạng gridlayout có kích thước 20x20



Với mỗi ô trong layout tương ứng với 1 nút bấm



Quy định lượt đánh trước là “o” , đánh số cho nó là số chẵn, lượt đánh sau là số lẻ Với mỗi lần xảy ra sự kiện trên 1 nút được tính là 1 nước đánh



Class LogicWin để kiểm tra xem điều kiện win đã thỏa mãn hay chưa

/\*\*

* Copyright(C) 2021 Luvina Software Company
* LogicWin.java, Aug 5, 2021 Đức Nguyễn

\*/

**package** controller;

**import** javax.swing.JButton;

**import** ui.MainPanel;

/\*\*

* **@author** Dell
* Kiểm tra điều kiện thắng cho 2 người chơi

\*/

**public class** LogicWin {

// Khởi tạo mảng các nút là các nút trên main panel được gắn các phương thức xử lý

JButton [][] btn = MainPanel.*btns*;

/\*\*

* + Hàm kiểm tra điều kiện thắng
  + **@param** i chỉ số của nút trên hàng ngang
  + **@param** j chỉ số của nút trên hàng dọc
  + **@param** text giá trị của nút hiện thời được trỏ đến
  + **@return** true hay false

\*/

**public boolean** checkWin(**int** i, **int** j, String text) {

**boolean** check = **false**; **int** dem=0;

// kiểm tra theo chiều dọc

// kiểm tra các phần tử xung quanh phần tử đang trỏ đến có

// giá trị giống với nó hay không

**for** (**int** k=-4; k<=4;k++) {

**if** (i+k >=0 && i+k<20) {

// với mỗi nút có giống trị giống

**if** (btn[i+k][j].getText() == text) {

// số lượng nút +1 dem++;

}

}

}

// nếu số lượng nút có giá trị giống nhau = 5

**if** (dem==5) {

// kiểm tra có win hay không check=**true**;

}

// nếu kiểm tra theo chiều dọc không thỏa mãn, đưa dem = 0 để

kiểm tra theo chiều dọc

dem=0;

// kiểm tra theo chiều ngang

// kiểm tra các phần tử xung quanh phần tử đang trỏ đến có

// giá trị giống với nó hay không

**for** (**int** k=-4;k<=4;k++) {

**if** (j+k>=0 && j+k<20) {

// với mỗi nút có giống trị giống

**if** (btn[i][j+k].getText() == text) {

// số lượng nút +1 dem++;

}

}

}

// nếu số lượng nút có giá trị giống nhau = 5

**if** (dem==5) {

// kiểm tra có win hay không check = **true**;

}

// nếu kiểm tra theo chiều ngang không thỏa mãn, đưa dem = 0 để

kiểm tra 2 đường chéo

dem=0;

// kiểm tra 2 đường chéo

// kiểm tra các phần tử xung quanh phần tử đang trỏ đến có

// giá trị giống với nó hay không

**for** (**int** k=-4, t=4; k<=4 && t>=-4; k++, j--) {

**if** (j+k >= 0 && j+k < 20 && i+k>=0 && i+k < 20) {

// với mỗi nút có giống trị giống

**if** (btn[i+k][j+t].getText() == text) {

// số lượng nút +1 dem++;

}

}

}

**if** (dem == 5) {

// kiểm tra có win hay không check = **true**;

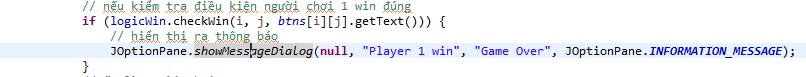
}

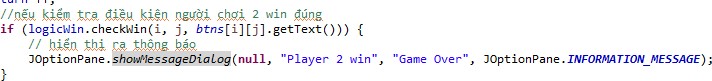
**return** check;

}

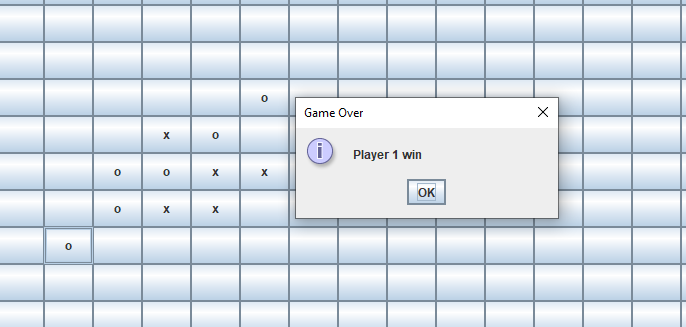
}

### Nếu điều kiện thắng được thỏa mãn thì hiển thị ra thông báo và kết thúc chương trình





Demo ứng dụng



## Tổng kết

* Hoàn thành nội dung thực tập tại công ty
* Kết thúc thực tập ngày 5/8/2021