CHƯƠNG I - BẮT ĐẦU VỚI JAVA

Bài 1 – Hello world

Bạn hãy xem bài của anh CEO trong JVN

Bài 2 – In ra chuỗi nhập vào

Bài đầu tiên của bạn, bạn đã học cách để Java in cái gì đó ra màn hình, trong bài này, bạn sẽ học cách nhập vào cái gì đó và Java in cái đó ra màn hình. Gõ cái này đi bạn (lưu ý, bạn phải gõ, không được copy và paste)

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
		System.out.print("Your name is: ");
		String str;
		str = in.readLine();
		System.out.println("Welcome " + str + " to Java");
	}
}
```

Xin hỏi, tôi đã bắt đầu với một vài ví dụ, mà tôi chẳng hiểu cái gì cả.
Xin trả lời, bạn sẽ học Java qua các ví dụ, rất nhiều ví dụ, lí thuyết thì bạn chỉ học từ từ thôi.

*Lí thuyết: cấu trúc một chương trình Java

```
public class Core {
	public static void main(String[] args) {
		System.out.println("Hello,Everybody in the World!");
	}
}
```

public class Core bạn bắt đầu một lớp Java
public static void main(String[] args) đây là một phương thức main trong Java, để cho chương trình chạy được. Tạm thời bạn phải gõ y như thế này
System.out.println("Hello,Everybody in the World!") đây là một câu lệnh trong Java, đơn giản nó chỉ in ra chuỗi nằm trong 2 dấu "" ra màn hình.
Mọi lớp và phương thức trong Java mở ra bằng { và đóng lại bằng }
Mọi câu lệnh trong java kết thúc bằng ;

Bài 3 – Biến trong Java

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
```

```
                System.out.print("Nhap a: ");
                int a = Integer.parseInt(in.readLine());
                System.out.print("Nhap b: ");
                int b = Integer.parseInt(in.readLine());
                int ketqua;
                ketqua = a+b;
                System.out.println("Ket qua bai toan a+b la: " + ketqua);
        }
}
```

Nhập thử 2 số a và b vào đi bạn, kết quả bài toán a+b sẽ được in ra.

*Lí thuyết:

```
import java.io.*;
public class Hello {
        public static void main(String[] args) throws Exception {
                BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
                System.out.print("Your name is: ");
                String str;
                str = in.readLine();
                System.out.println("Welcome " + str + " to Java");
        }
}
```

Tạm thời, trong chương trình này, bạn chỉ nên quan tâm đến dòng
String str khai báo biến str kiểu chuỗi, và
System.out.println("Welcome " + str + " to Java")
Đây cũng là dòng System.out.println như chương trình đầu, có khác là + str + tức là đưa một biến vào chuỗi in ra. Chỉ đến đó thôi nhé, sau đó, hãy quan tâm đến bài hôm nay
System.out.print("Nhap a: ");
int a = Integer.parseInt(in.readLine());
System.out.print("Nhap b: ");
int b = Integer.parseInt(in.readLine());
int ketqua;
ketqua = a+b;
System.out.println("Ket qua bai toan a+b la: " + ketqua);

*Giải thích
import bạn nhập class hay thư viện chuẩn, tạm thời đừng quan tâm nó là gì, chỉ cần nhớ là có nó để chương trình chạy
System.out.print in ra một chuỗi, nhưng không xuống dòng
System.out.println in ra một chuỗi, nhưng xuống dòng
int ketqua tức là khai báo biến ketqua kiểu int

ketqua = a+b tức là gán kết quả một biểu thức tính toán (ở đây là biến a + biến b) cho biến ketqua
System.out.println("Ket qua bai toan a+b la: " + ketqua) thì đơn giản rồi, in cái dòng đó ra, chỉ khác là nó đưa biến ketqua của bạn vào chuỗi đó.

Bài 4 – Chia hết, chia lấy dư

*Lí thuyết: một số kiểu biến trong Java

Bạn đã biết 2 kiểu String (chuỗi) và int (nguyên) bây giờ bạn biết thêm kiểu float (thực) Số nguyên và số thực bạn biết sự khác nhau rồi chứ. Bây giờ ta bắt đầu bài toán ví dụ

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
		System.out.print("Nhap a: ");
		float a = Float.parseFloat(in.readLine());
		System.out.print("Nhap b: ");
		float b = Float.parseFloat(in.readLine());
		float ketqua = a/b;
		System.out.println("Ket qua bai toan a+b la: " + ketqua);
	}
}
```

Bạn thử bài toán xem, nhớ đừng nhập số b=0 nhé, chuyện ấy sẽ xử lí sau.
Ví dụ nhập a=5, b=2, kết quả in ra sẽ là 2.5, thú vị phải không ?

Bây giờ cũng bài toán ấy, bạn thay đổi như sau:

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
		System.out.print("Nhap a: ");
		int a = Integer.parseInt(in.readLine());
		System.out.print("Nhap b: ");
		int b = Integer.parseInt(in.readLine());
		float ketqua = a/b;
		System.out.println("Ket qua bai toan a+b la: " + ketqua);
	}
}
```

Cũng nhập a=5, b=2, lần này kết quả in ra là … 2
Phép chia sẽ là phép chia hết nếu cả 2 toán hạng đều kiểu nguyên, gọi là chia lấy nguyên (/) hay div

Bây giờ cũng chương trình ấy mà ta thay đổi lại chút xíu xem sao

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
		System.out.print("Nhap a: ");
		int a = Integer.parseInt(in.readLine());
		System.out.print("Nhap b: ");
		int b = Integer.parseInt(in.readLine());
		float ketqua = a%b;
		System.out.println("Ket qua bai toan a+b la: " + ketqua);
	}
}
```

Cũng nhập a=5, b=2, lần này kết quả in ra là … 1
Đây là kết quả phép chia lấy dư 5 chia cho 2, gọi là chia lấy dư (%) hay mod

*Thế nếu tôi muốn 2 số nguyên chia nhau mà ra kiểu thực chứ không phải phép chia lấy nguyên thì sao ? Trong trường hợp đó, bạn dùng "ép kiểu"
int a=5,b=2;float ket qua;
ketqua=(float)a/b;

Bài 5 – Lập trình OOP
Bạn xem bài của anh CEO trong JVN
class
Đây là một class, class này có hai property (thuộc tính) là name và age

```
public class Person
{
	String name;
	int age;
}
```

Đây là một class, class này ngoài property còn có constructor (khởi tạo) của class đó

```
public class Person
{
	String name;
	int age;
	public Person(String name)
	{
		this.name = name;
	}
}
```

Trong cái constructor này hãy lưu ý một điều, đó là biến this. Biến this có nghĩa là bản thân cái class đó (ở đây là class Person).

Trong class Person có một property là age, câu this.age = age có nghĩa là cái thuộc tính age của class Person sẽ nhận giá trị ở cái đối số age do constructor Person(int age) đưa vào.
Lưu ý là mọi class đều có sẵn ít nhất một constructor không có đối số.
Đây là một class, class này ngoài property, constructor còn có một behavior (hành vi)

```
public class Person
{
        String name;
        int age;
        public Person(int age)
        {
                this.age = age;
        }
        public void Nhap()
        {
                nameonsole.readLine("Nhap ho ten:");
        }
}
```

Khi ta viết câu lệnh sau
Person personOne = new Person(12);
Thì ta đã tạo ra một instance (thể hiện) là personOne của class Person

Khai báo một class

public abstract class MyClass {}

Từ thứ 1 là khai báo quyền truy xuất và kế thừa,có 3 loại
-public:được phép truy xuất từ bất cứ nơi nào và bất cứ lớp nào cũng được quyền kế thừa
-protected:chỉ có phương thức cùng gói được phép truy xuất và kế thừa
-private:chỉ có phương thức cùng gói được phép truy xuất nhưng không lớp nào được phép kế thừa
-nếu không khai báo,mặc định là protected

Từ thứ 2 là khai báo một lớp trừu tượng hay là không trừu tượng

Nhiem vu: tao 1 lop Person, tao tiep 2 lop Students va Teachers ke thua lop Person, tao lop Execute chua ham chinh de chay chuong trinh.

--lop Person-

```
import corejava.*;
abstract class Person
{
        //cai nay goi la cac property hay state-thuoc tinh cua doi tuong
        String hoten;
        int age;
```

```
        String diachi;
        int luong;
        //cac constructor
        public Person(int age)
        {
                this.age = age;
        }
        //cac method hay behavior-hanh vi cua doi tuong
        public void Nhap()
        {
                hoten = Console.readLine("Nhap ho ten:");
                diachi = Console.readLine("Nhap dia chi:");
        }
        //vi la 1 class thuoc loai abstract nen Person duoc phep khai bao cac
method khong co noi dung, noi dung cua class In se duoc cac lop ke thua no
them vao noi dung cua rieng no
        public abstract void In();
        public abstract int Tinhluong();
}
```

--lop Students-

```
import corejava.*;
class Students extends Person
{
        int MaSV,Malop;
        public void Nhap()
        {
                super.Nhap();
                MaSV = Console.readInt("Nhap ma SV:");
                Malop = Console.readInt("Nhap ma lop:");
        }
        public void In()
        {
                System.out.println(hoten);
                System.out.println(diachi);
                System.out.println(MaSV);
                System.out.println(Malop);
        }
        public int Tinhluong()
        {
        return 150000;
        }
}
```

tu khoa super se goi ham Nhap() tu lop Person la cha cua lop Students

--lop Teachers-

```
import corejava.*;
class Teachers extends Person
{
        int Makhoa;
        public void Nhap()
        {
                super.Nhap();
                Makhoa = Console.readInt("Nhap ma khoa::");
        }
```

```
        public void In()
        {
                System.out.println(hoten);
                System.out.println(diachi);
                System.out.println(Makhoa);
        }
        public int Tinhluong()
        {
                return 500000;
        }
}
```

--lop Execute-

```
import corejava.*;
class Execute
{
        public static void main(String args[])
        {
                Students st = new Students();
                st.Nhap();
                st.In();
                st.luong=st.Tinhluong();
                Teachers tc = new Teachers();
                tc.Nhap();
                tc.In();
                tc.luong=tc.Tinhluong();
        }
}
```

Khai báo một thuộc tính:

public static void temp;

Từ thứ 1 là khai báo quyền truy xuất,có 3 loại
-public:được phép truy xuất từ bất cứ nơi nào
-protected:chỉ có lớp con mới được phép truy xuất
-private:chỉ có lớp đó xài(thuộc tính riêng của nó)
-nếu không khai báo,mặc định là protected

Từ thứ 2 là khai báo cách truy xuất static(tĩnh)

-nếu không khai báo,mặc định là không tĩnh
Tất cả các đối tượng thể hiện từ lớp cha đều được phép thay đổi giá trị của các thuộc tính không tĩnh,còn giá trị của thuộc tính tĩnh thì không được phép thay đổi

```
public class Car
{
        public string branch;
        public int cost;
```

```
        public static int tire=4;
}
```

Như ví dụ trên,tất cả các lớp con của lớp Car (như ToyotaCar,Peugeot,Mazda…) đều được phép thay đổi các thuộc tính branch hay cost để phù hợp cho riêng mình,nhưng thuộc tính tire (số bánh xe) không được phép thay đổi vì là thuộc tính tĩnh
Nói cách khác, chỉ có một và chỉ một thuộc tính có tên là tire trong class Car và tất cả các class con của nó, vì vậy gọi là tĩnh

Khai báo một hành vi
Một phương thức được khai báo như sau
public static double ketqua()

Có 3 chỉ định truy xuất là public, protected và private
-public:được phép truy xuất từ bất cứ nơi nào
-protected:chỉ có lớp kế thừa lớp chứa nó được truy xuất
-private:chỉ lớp chứa nó được truy xuất(dùng nội bộ)
-nếu không khai báo,mặc định là protected

Có 6 chỉ định thuộc tính là static, abstract, final, native, synchronized (đồng bộ) và volatile (linh hoạt), static(tĩnh)

-nếu không khai báo,mặc định là không tĩnh

```
class TestObject
{
        static void StaticMethod() {…}
        void NonStaticMethod() {…}
}
```

Nếu là một phương thức không tĩnh, đầu tiên bạn phải khởi tạo một đối tượng,sau đó mới được phép gọi phương thức
TestObject test=new TestObject();
test.NonStaticMethod();
Nếu là một phương thức tĩnh,bạn được phép gọi trực tiếp từ lớp
TestObject.StaticMethod();
abstract(trừu tượng)
Một phương thức trừu tượng không có nội dung.Nội dung của nó sẽ được các lớp con tùy biến và phát triển theo hướng của riêng nó.
- final: không thể được extends hay override (ghi đè)
- native: thân phương thức viết bằng C hay C++
- synchronized: chỉ cho phép 1 thread truy cập vào khối mã ở cùng một thời điểm
- volatile: sử dụng với biến để thông báo rằng giá trị của biến có thể được thay đổi vài lần vì vậy không ghi vào thanh ghi

Từ thứ 3 là giá trị trả về.Nếu không có giá trị trả về thì là void

Interface-template

Bây giờ ta có 1 khái niệm mới, là giao diện. Giao diện ra đời chính là để giải quyết đa kế thừa. Mỗi lớp trong Java chỉ có 1 lớp cha, nhưng có thể implements nhiều giao diện. Giao diện được khai báo giống như 1 lớp, cũng có state và behavior. Nhưng state của giao diện là final còn behavior là abstract
Giả sử, ta sẽ khai báo một giao diện

```
public interface Product
{
        //hai state duoi day la final, tuc la lop implements khong duoc phep
doi gia tri
        static string maker = "My Corp";
        static string phone = "555-7767";
        //behavior duoi day la abstract, tuc la khong co noi dung
        public int getPrice(int id);
}
```

Bây giờ, ta sẽ viết một class có cài đặt (implements) giao diện này

```
public class Shoe implements Product
{
        public int getPrince(int id)
        {
                return (id= =1)?5:10;
        }
        public String getMaker()
        {
                return maker;
        }
}
```

Muốn implements nhiều giao diện, làm như sau, ví dụ class Toyota extends Car implements ActionCar, ActionMobilation

package-unit
Hãy tạo 1 thư mục có tên là Transport
Bên trong thư mục này hãy tạo 2 file là Car.java và Bicycle.java như sau

--Car.java-

```
package Transport;
public class Car
{
        public String manufacturer;
        public int year;
}
```

--Bicycle.java-

```
package Transport;
public class Bicycle
{
        public int cost;
        public Bicycle(int cost)
```

```
        {
                this.cost = cost;
        }
}
```

Như vậy là ta đã tạo ra 1 gói chứa 2 lớp là Car và Bicycle. Bây giờ ta có 1 chương trình muốn sử dụng gói này là TestProgram.java. Ta viết:

--ViDuTransport.java-

```
import Transport.*;
class TestProgram
{
        public static void main(String args[])
        {
                Car myCar = new Car();
                myCar.manufacturer = "Toyota";
                Bicycle myBicycle = new Bicycle(1500);
        }
}
```

Lưu ý nếu trong file ViDuTransport bạn không khai báo import Transport.* thì bạn vẫn có thể khai báo tường minh như sau
Transport.Car myCar = new Transport.Car();

nạp chồng (overload) 1 phương thức

```
class Vidu
{
        public satic void main(String a[])
        {
                private  float cost;
                public float CalculateSalePrice()
                {
                        return cost*1.5;
                }
                public float CalculateSalePrice(double heso)
                {
                        return cost*(1+heso);
                }
        }
}
```

Ở đây có 2 phương thức trùng tên CalculateSalePrice nhưng phương thức thứ 2 khác tham số, gọi là nạp chồng
* nạp chồng (overload) và ghi đè (override)
Những phương thức được nạp chồng là những phương thức trong cùng một lớp, có cùng một tên nhưng danh sách đối số khác nhau
Phương thức được ghi đè là phương thức có mặt ở lớp cha, được xác định là phương thức chung cho các lớp con, rồi xuất hiện ở các lớp con

Nạp chồng là một hình thức đa hình (polymorphism) trong quá trình biên dịch (compile) còn ghi đè là trong quá trình thực thi (runtime)

Bài 6 – Các kiểu dữ liệu nguyên thủy và phép toán

- Kiểu nguyên: gồm số nguyên(int,long)
- Kiểu dấu phẩy động (hay kiểu thực): gồm số thực(float,double)
- Kiểu kí tự (char)
- Kiểu chuỗi (String)

Hằng kí tự khai báo như sau, ví dụ 'H' (khác với "H" là một chuỗi kí tự)
Một số hằng kí tự đặc biệt, ví dụ '\\' để biểu diễn chính kí tự \, và \u biểu diễn Unicode, ví dụ:
'\u00B2' biểu diễn ² (bình phương)
'\u00BC' biểu diễn ¼ (một phần tư)
'\u0170' biểu diễn ª (mũ a)
- Kiểu boolean
Có 2 giá trị là 2 từ khóa true và false, và không thể chuyển kiểu sang int

*Khai báo biến
int i,j; //2 biến i và j có kiểu dữ liệu là int
char ch='A'; //biến ch kiểu char khởi tạo giá trị đầu 'A'

*Khai báo hằng
Hằng được khai báo với từ khóa final. Ví dụ:
final float PI = 3.14159;

*Phép toán
Phép toán của Java giống C. Trong class java.lang.Math có một số method để dùng trong toán học như sau
double y = Math.pow(x,a) = x^a
và random, sin, cos, tan, exp (mũ), log(logarit) ...

* Các phép toán số học
- Với cả kiểu nguyên và kiểu thực: + - * / (phép chia sẽ cho ra kết quả kiểu thực nếu một trong 2 toán tử là kiểu thực)
- Chia hết (/) chỉ áp dụng khi cả 2 toán tử là kiểu nguyên, ví dụ 10/3=3

- Chia lấy dư (%) chỉ áp dụng khi cả 2 toán tử là kiểu nguyên, ví dụ 10%3=1

* Các phép toán quan hệ (so sánh)
- Bao gồm ==,<,>,<=,>= trả về kiểu boolean

* Các phép toán với kiểu logic
- Bao gồm and(kí hiệu &&) or(kí hiệu ||) not(kí hiệu !)

* Phép ++ và --
- Phép này có 2 dạng, một là ++biến hay --biến, hai là biến++ hay biến-- Sự khác nhau chỉ là khi phép này thực hiện chung với một phép toán khác thì
- Với ++biến và --biến thì nó sẽ thực hiện phép toán này trước rồi mới thực hiện phép toán khác
- Với biến++ và biến-- thì nó sẽ thực hiện phép toán khác trước rồi mới thực hiện phép toán này

* Phép gán
- Phép này có dạng a=5
- Phép gán phức, ví dụ a+=5 nghĩa là a=a+5, hay a*=2 nghĩa là a=a*2

* Trình tự kết hợp
Hầu hết các phép toán điều có trình tự kết hợp từ trái sang phải, chỉ có các phép sau là từ phải sang trái
- Phép ++ và --
- Các phép gán như =,+=,-=,<<=,>>=

Bài 7 – Mệnh đề if

nếu em đẹp thì tôi sẽ cưa em không thì tôi cưa đứa khác
IF em đẹp THEN tôi sẽ cưa em ELSE tôi cưa đứa khác
IF(em đẹp) tôi sẽ cưa em;
ELSE tôi cưa đứa khác;

Cú pháp (syntax) của mệnh đề IF là
if(mệnh đề) lệnh 1;
else lệnh 2;
Nếu mệnh đề đúng thì thực hiện lệnh 1;

Không thì thực hiện lệnh 2;

Ví dụ
if(a>b) System.out.println("So lon nhat la "+a);
else System.out.println("So lon nhat la "+b);

*Ta xây dựng một bài toán làm tròn số
Nhập vào một số bất kì. Nếu phần thập phân số này >=0.5, làm tròn tăng lên một đơn vị, ngược lại giảm đi một đơn vị.

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
		System.out.print("Nhap a: ");
		float a = Float.parseFloat(in.readLine());
		float ketqua=a%1;
		if(ketqua>=0.5) a=a-ketqua+1;
		else a=a-ketqua;
		System.out.println("Ket qua bai toan la: " + a);
	}
}
```

* Phép điều kiện ? và phép chọn :
- Giả sử có mệnh đề if
if(a>b) a=2;
else a=0;
Phép điều kiện biểu diễn như sau a=a>b?2:0 nghĩa là nếu chân trị của a>b là đúng thì a=2 nếu là sai thì a=0

* Sau khi học xong if, bạn có rất nhiều bài tập để mà … làm, cổ điển nhất vẫn là giải phương trình bậc một và hai, ngoài ra còn nhiều bài tập khác nữa. Ở đây chỉ có giải phương trình bậc một. Bạn nên tìm nhiều bài tập để tự làm trước khi tiếp tục phần kế.

Ví dụ: phương trình bậc 1

```
import java.io.*;
public class Hello {
	public static void main(String[] args) throws Exception {
		BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
		System.out.println("Giai phuong trinh bac nhat dang ax+b=0");
		System.out.print("Nhap he so a: ");
		float a = Float.parseFloat(in.readLine());
		System.out.print("Nhap he so b: ");
		float b = Float.parseFloat(in.readLine());
		if(a==0) {
			if(b==0) System.out.println("Phuong trinh vo so
nghiem");
			if(b!=0) System.out.println("Phuong trinh vo dinh");
		}
		else System.out.println("Phuong trinh mot nghiem x=" + -b/a);
```

```
        }
}
```

Bài 8 – switch

Bạn đã học xong if. Bạn muốn dùng vòng lặp if để đánh giá điểm số nhập vào. Bạn sẽ viết chương trình sau đây

```
import java.io.*;
public class Hello {
        public static void main(String[] args) throws Exception {
                BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
                System.out.print("Nhap diem so: ");
                int diem = Integer.parseInt(in.readLine());
                if(diem<=2) System.out.println("Yeu");
                if((diem>2) && (diem<=3)) System.out.println("Trung binh");
                if((diem>3) && (diem<=4)) System.out.println("Kha");
                if((diem>4) && (diem<5)) System.out.println("Gioi");
                if(diem==5) System.out.println("Xuat sac");
        }
}
```

Thay vì lặp lại những câu if ấy, bạn nên dùng switch

```
import java.io.*;
public class Hello {
        public static void main(String[] args) throws Exception {
                BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
                System.out.print("Nhap diem so: ");
                int diem = Integer.parseInt(in.readLine());
                switch(diem)
                {
                        case 0:
                        case 1:
                        case 2:System.out.println("Yeu");break;
                        case 3:System.out.println("Trung binh");break;
                        case 4:
                        case 5:System.out.println("Gioi");break;
                        default:System.out.println("Vao sai");
                }
        }
}
```

*break với switch: break sẽ thoát ngay ra khỏi khối lệnh trong thân của switch

Bài 9 – String

Khác với C, String là một lớp của Java. String được khai báo như sau
String a = "Hello";
Cộng 2 String bằng dấu +
System.out.println("Gia tri la " + n);
Java có khả năng tự chuyển kiểu bất cứ dữ liệu kiểu số nào khi cộng vào String. Dù n là int, float, double đều có thể chuyển thành String nhờ mẹo vặt ("" + n)

Các method trong class String
* substring
String s1 = "Hello";
String s2 = s1.substring(0,4); //bắt đầu từ kí tự thứ 0 (tức là 'H') lấy đi 4 kí tự (tức là "Hell")

* length
int n = s1.length(); //tức là bằng 5

* charAt
char ch = s1.charAt(4); //tức là bằng 'o'
Đây là method tìm kí tự thứ i trong String, các kí tự trong String được đánh số từ 0

* equals
Kiểu tra xem chuỗi nguồn s có giống chuỗi đích d hay không, ta dùng method equals trả về boolean
boolean b = s.equals(t);
String không giống dữ liệu kiểu số, tuyệt đối không dùng giống như if(s==t)

* compareTo
int a = s2.compareTo(s1);
a>0 s2>s1
a<0 s2<s1
a=0 s2=s1
So sánh giữa s2 và s1 là so sánh thứ tự giữa kí tự đầu của hai chuỗi so đi, ví dụ "kc" > "kazbe"

* toCharArray (đổi chuỗi ra mảng kí tự)
char[] chuoi = s1.toCharArray();

* indexOf
String s1 = "Hello Everybody";
String s2 = "lo";
int n = s1.indexOf(s2); //n sẽ bằng 4
Đây là method trả về vị trí của chuỗi s2 trong chuỗi s1, nếu không tìm thấy sẽ trả về -1

* Chuyển kiểu từ String ra dữ liệu kiểu số
Chuyển từ dữ liệu kiểu số ra String khá dễ dàng, dùng "" + n, nhưng ngược lại thì phải dùng các method tương ứng.
Các method này nằm trong góijava.lang, trong các class Byte, Short, Integer, Long, Float, Double
String input = "230";
int n = Integer.parseInt(input); //n sẽ bằng 230
Tương tự với các method sau Byte.parseByte, Short.parseShort, Float.parseFloat, ...

Bài 10 – vòng lặp for

```
for(int i=0;i<n;i++)
        s+=i;
```

3 thành phần trong câu for ta có thể bỏ hết nhưng phải giữ lại các dấu ; khi đó nếu muốn ta có thể đặt phép toán điều khiển vòng lặp trong thân lệnh như sau

```
for(int i=0;i<n;)
{
        s+=i;
        i++;
}
```

*break với for: break sẽ thoát ngay ra khỏi vòng for

```
for(int i=0;i<n;i++)
{
        System.out.println(i);break;
        System.out.println("Tiep tuc");
}
```

Kết quả in ra không có câu "Tiep tuc" vì break nhảy ngay ra khỏi vòng for sau khi in 1
*continue với for: continue sẽ khiến vòng for bắt đâu 1 chu trình mới và bỏ qua tất cả các lệnh bên dưới nó

VD: in tất cả các số từ 0 đến 10, bỏ qua 3,4,5

```
for(int i=0;i<10;i++)
{
        if((i= =3)||(i= =4)||(i= =5)) continue;
        System.out.println(i);
}
```

Bài 11 – while

while(biểu thức)
	lệnh;

Nếu biểu thức đúng thì thực hiện lệnh

*break với while: break sẽ thoát ngay ra khỏi vòng while

```
int i=0;
while(i<10)
{
	System.out.println(i);break;
	i++;
}
```

Sẽ chỉ in ra 0

* continue với while: nó sẽ xác định giá trị biểu thức viết ngay sau while

```
int i=0;
while(i<10)
{
	System.out.println(i);continue;
	i++;
}
```

Ta sẽ được một loạt in 0 vô tận

Bài 12 – vòng lặp do..while

do
	lệnh;
while(biểu thức);

Nếu biểu thức đúng thì tiếp tục thực hiện lệnh

*break với do..while: break sẽ thoát ngay ra khỏi vòng while

```
int i=0;
do
{
	System.out.println(i);break;
	i++;
}
```

while(i<10);

Sẽ chỉ in ra 0

* continue với while: nó sẽ xác định giá trị biểu thức viết ngay sau

```
while
int i=0;
do
{
	System.out.println(i);continue;
	i++;
}
while(i<10);
```

Ta sẽ được một loạt in 0 vô tận

Bài 13 – array

Ta khai báo 1 mảng với câu lệnh sau, và không cung cấp số phần tử
int[] a;
Tuy vậy, với Java, để dùng được một array, ta cần phải khởi tạo array đó, và lúc này phải cung cấp số phần tử
int[] a;
a = new int[100];
Hai câu có thể viết lại thành một câu
int[] a = new int[100];
Java sẽ khởi tạo một mảng 100 phần tử đều là int có đánh thứ tự từ 0 đến 99
Mảng có giá trị đầu: Mảng loại này không cần new mà cũng chẳng cần số phần tử
int[] a = {1,45,6,8,21};

Các method với mảng
* length
method này sẽ cung cấp số phần tử của mảng, ví dụ ta muốn gán giá trị số cho các phần tử của mảng a
for(int i=0;i<a.length;i++) a|i|=i; lưu ý là length, không phải length()

*System.arraycopy
Giả sử, ban đầu ta có 2 mảng
int[] s = {1,3,5,7,9,11,13,15};
int[] d = {2,4,6,8,10,12,14};
method arraycopy trong gói System
System.methodcopy(s,3,d,2,4);
sẽ cho ra một mảng d mới là {2,4,7,9,11,13,14}
method này sẽ thay thế 4 phần tử, tính từ phần tử thứ 2 trong mảng d, bằng ngần ấy phần tử tính từ phần tử thứ 3 trong mảng s

Các method nằm trong class java.util.Arrays
* void sort
Nó sẽ sắp xếp một mảng số tăng dần
int[] s = {28,7,14,11};

Arrays.sort(s);

* int binarySearch
Nó sẽ tìm vị trí của một phần tử trong một mảng, trả về -1 nếu không tìm thấy
int[] s = {28,7,14,11};
int n = Arrays.binarySearch(s,14); n sẽ bằng 2

Mảng nhiều chiều
int[][] = new int[100][50];
Hoặc khai báo 1 mảng có giá trị đầu. Đây là mảng 2 chiều gồm 4 phần tử là 4 mảng 1 chiều, mỗi mảng 1 chiều chứa 3 phần tử
int[][] a =
{
{16, 3, 2},
{5, 10, 11},
{9, 6, 7},
{4, 15, 14}
};

Bài 14 - ngoại lệ

```
int x,y;
x=10;y=x-10;
x=x/y;
```

Khi chạy đoạn mã này bạn sẽ thấy xuất hiện thông báo
java.lang.ArithmeticException: divide by zero
Và chương trình sẽ thoát ra ngay lúc đó. Muốn chương trình chạy tiếp và không thoát ra, ta đón "bắt" ngoại lệ này, đưa ra biến e, cuối cùng in e (để xem là ngoại lệ gì)

```
int x,y;
try
{
        x=10;y=x-10;
        x=x/y;
}
catch(Exception e)
{
        System.out.println(e.getMessage());
}
```

Xử lí ngoại lệ (Exception)

Để "ném" ngoại lệ do bất cứ dòng mã nào trong một phương thức sinh ra, bạn có thể khai báo để ném bỏ ngoại lệ đó

```
public void divide() throws Exception
{
```

```
        int a=5/0;
}
```
hoặc nếu muốn "bắt" ngoại lệ đó lại để xem đó là ngoại lệ gì để xử lí, bạn "bắt" nó rồi in ra
```
try
{
        int a=5/0;
}
catch(Exception e)
{
        System.out.println(e.getMessage());
}
```

Nếu muốn chương trình thành công thì sinh thông báo thành công, thất bại thì sinh thông báo ngoại lệ, bạn có thể dùng
```
boolean done=false;
try
{
        int a=5/b;
        done=true;
}
catch(Exception e)
{
        System.out.println(e.getMessage());
}
if(done==true) System.out.println("Successful");
```

Bài 15 - Vector (mảng không giới hạn số phần tử)

Các method trong bài này nằm ở 2 class java.util.Vector và java.util.Enumeration
Khai báo
Vector vt = new Vector();
Nhập dữ liệu cho một Vector (class Console nằm trong gói corejava)
Lưu ý là mỗi phần tử của Vector đều phải là một đối tượng, nên ta phải có new Integer(n) khi muốn đưa vào một biến kiểu int. Tương tự với Byte, Long, Float, ...
```
do
{
int n = Console.readInt("");
if(n!=0) vt.addElement(new Integer(n));
}
while(n!=0);
In ra các phần tử của một Vector
for(int i=0;i<vt.size();i++)
System.out.println(vt.elementAt(i));
```

Để đưa Vector về kiểu mảng cho dễ thao tác, ta đưa về kiểu Enumeration (một kiểu mảng)

Enumeration e = vt.elements();
Như vậy ta có mảng e kiểu Enumeration sao chép y khuôn Vector vt để dễ xử lí, không đụng đến Vector vt
In ra các phần tử của một Enumeration

```
while(e.hasMoreElements())
System.out.println(e.nextElement());
```

Bài 16 - Lớp nội (lớp nằm trong lớp khác)

```
public class TestProgram
{
        static int currentCount;
        static class Apple
        {
                int weight;
                public Apple(int weight)
                {
                        this.weight=weight;
                        currentCount++;
                }
                public int Weight()
                {
                        return weight;
                }
        }
        public static void main(String args[])
        {
                Apple a=new Apple(12);//khoi tao 1 quả tao nang 12kg
                System.out.print(a.Weight());
        }
}
```

Ở đây ta thấy lớp nội Apple trong lớp TestProgram, khi biên dịch Java sẽ làm xuất hiện 2 file là TestProgram.class và TestProgram$Apple.class. Ưu điểm khi sử dụng lớp nội là:
- thể hiện tính đóng gói cao
- các lớp nội có thể truy xuất trực tiếp các biến của lớp cha
Lưu ý là lớp nội khác với các lớp mà nằm chung một file, ví dụ như tập tin MainClass.java dưới đây

```
public class MainClass
{
}
class Subclass
{
}
```

Bài 17 - Tạo tập tin jar tự chạy

Giả sử chương trình của bạn có vài file .class trong đó file chương trình chính là MainPro.class

chẳng hạn.
Bạn hãy tạo một file lấy tên là mymf.mf có nội dung như sau
Main-Class: MainPro

Bắt buộc phải chính xác như thế (tức là phải có cả xuống dòng), không thì trình chạy jar không hiểu được.
Sau đó bạn vào %JAVA_HOME%\bin\ chép tất cả các tập tin .class của ứng dụng và cả mymf.mf vào đó, rồi chạy jar.exe với tham số dòng lệnh như sau
jar cmfv mymf.mf MyProgram.jar *.class
Tương tự nếu bạn muốn đưa thêm 2 thư mục dir1 và dir2 vô file JAR thì bạn cũng gõ
jar cmfv mymf.mf MyProgram.jar *.class dir1 dir2
Trình jar sẽ tạo file MyProgram.jar (tên khác tùy bạn) có thể chạy được, không phải dùng lệnh java hay giả sử không có IDE quen thuộc của bạn

CHƯƠNG 2 - JAVA VÀ LẬP TRÌNH GIAO DIỆN BẰNG SWING

Đã đến lúc bạn nên sử dụng một IDE để công việc của mình nhanh chóng và dễ dàng hơn. Applet đã trở thành đồ cổ, chúng ta nhảy luôn sang AWT – Swing

Bài 1 - Mở đầu về Swing

Chương trình này sẽ tạo một JFrame đơn giản nhất

```
import javax.swing.JFrame;
class HelloWorldSwing
{
	public static void main(String[] a)
	{
		JFrame frame=new JFrame("Main Frame");//Main Frame la ten cai
cua so
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);//ham
dong cua so lai
		JLabel label=new JLabel("Hello Everybody, label contain
context");//mot doi tuong do hoa
		frame.getContentPane().add(label);//dua doi tuong do hoa vao
trong frame
		frame.pack();//"dong goi" lai toan bo trinh do hoa
		frame.setVisible(true);//hien thi trinh do hoa ra man hinh
	}
}
```

Đây là một Frame đơn giản khác, nhưng có thể dùng dễ dàng cho việc mở rộng chương trình

```
import javax.swing.JFrame;
import java.awt.*;
class Execute extends JFrame
{
	Container container = getContentPane();
```

```
        public Execute(String title)
        {
                super(title); //tuong duong JFrame(title)
                Label label=new Label("Hello Everybody, label contain
context");
                container.add(label);
        }
      public static void main(String a[])
      {
                Execute exe = new Execute("Frame");
                exe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                exe.pack();
                exe.setVisible(true);
        }
}
```

Hỗ trợ tiếng Việt

Giả sử bạn muốn nút bấm của bạn có dòng "Việt Nam" và bạn không biết in như thế nào, chương trình sau sẽ giúp bạn

JButton b=new JButton("Vi\u1EC7t Nam");

\u1EC7 là mã Unicode của kí tự ệ mà Java hỗ trợ. Tất cả kí tự Việt đều được hỗ trợ trong Latin và Latin Extend

Lưu ý là chỉ có javax.swing mới hỗ trợ, java.awt không hỗ trợ

Bài 2 - Cài đặt bộ nghe và sự kiện cho các đối tượng đồ họa

Các đối tượng đồ họa sở dĩ có thể hoạt động được là nhờ có các bộ nghe "nghe" các hành động mà người dùng tương tác với chuột hay bàn phím, và từ đó cho ra các sự kiện tương ứng.

Trong ví dụ dưới đây ta có class EventQuit là một bộ nghe, bộ nghe này thực hiện phương thức actionPerformed chính là chứa những sự kiện của bộ nghe đó. Đối tượng eventQuit là một instance của class EventQuit. Để cài đặt bộ nghe này cho đối tượng đồ họa button ta dùng phương thức addActionListener.

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core extends JFrame
{
      Container container = getContentPane();
      public Core(String title)
      {
                super(title);
                Button button = new Button("My button");
                EventQuit eventQuit=new EventQuit();
                button.addActionListener(eventQuit);
                container.add(button);
      }
      public static void main(String a[])
      {
```

```
            Core exe = new Core("Frame");
            exe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            exe.pack();
            exe.setVisible(true);
      }
        class EventQuit implements ActionListener
        {
                public void actionPerformed(ActionEvent e)
                {
                        System.exit(0);
                }
        }
}
```

Bây giờ, nếu ta muốn rút gọn, cài đặt bộ nghe và hành động trực tiếp, ta làm như sau

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {
                JFrame frame = new JFrame("My frame");
                final JButton button = new JButton("My button");
                button.addActionListener(new ActionListener()
                {
                        public void actionPerformed(ActionEvent e)
                        {
                                if(e.getSource()==button) System.exit(0); //nếu
event này có source do button sinh ra
                        }
                });
                frame.add(button);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.pack();
                frame.setVisible(true);
        }
}
```

Bài 3 - setLayout(null) đi đôi với setBounds

setLayout mạnh nhất trong Swing là setLayout(null) cho đối tượng add, còn đối với đối tượng bị add thì setBounds, cú pháp setBounds(x,y,width,height)
Trên monitor, Java tính điểm có tọa độ (0,0) là điểm trái trên cùng. Sau đó trục hoành (x) là chiều ngang monitor từ trái sang phải và trục tung (y) là chiều dọc monitor từ trên xuống dưới
Phương thức này sẽ tạo ra một hình chữ nhật ảo bao quanh đối tượng bị add, hình chữ nhật này có tọa độ góc đầu tiên là (x,y) và dài width cao height. Ví dụ như bài sau:

```
import javax.swing.JFrame;
import java.awt.*;
class Core
{
        public static void main(String args[])
```

```
        {
                JFrame frame = new JFrame("My frame");
                frame.setLayout(null);
                JButton b1 = new JButton("Button 1");
                b1.setBounds(0,0,100,25);
                frame.add(b1);
                JButton b2 = new JButton("Button 2");
                b2.setBounds(100,0,100,25);
                frame.add(b2);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setVisible(true);
        }
}
```

Như vậy là chúng ta đã setLayout(null) cho frame và lần lượt setBounds (kích thước cũng như vị trí) cho 2 button. Vậy để set kích thước cũng như vị trí cho chính frame thì dùng 2 phương thức sau:

```
import javax.swing.JFrame;
import java.awt.*;
class Core
{
        public static void main(String args[])
        {
                JFrame frame = new JFrame("My frame");
                frame.setLayout(null);
                JButton b1 = new JButton("Button 1");
                b1.setBounds(0,0,100,25);
                frame.add(b1);
                JButton b2 = new JButton("Button 2");
                b2.setBounds(100,0,100,25);
                frame.add(b2);
                frame.setLocation(200,100);
                frame.setSize(200,60);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setVisible(true);
        }
}
```

Bài này y chang bài trên, có khác là giờ đây vị trí điểm đầu của frame đã được xác định bằng setLocation, Nếu không setLocation, mặc định là (0,0) còn kích thước được xác định bằng setSize. Lưu ý là 200=chiều dài 2 cái button cộng lại còn 60=chiều rộng button + chiều rộng thanh ban đầu (=35).

Bài 4 - setLayout không phụ thuộc phân giải màn hình

Với các ứng dụng nhỏ thì chưa cần quan tâm lắm. Với các ứng dụng trung bình và lớn thì ứng dụng "co giãn" tùy theo độ phân giải sẽ là lợi thế lớn. Ta có thể lấy độ phân giải hiện hành và tùy biến ứng dụng như sau:

```
import javax.swing.JFrame;
import java.awt.*;
```

```
public class Core
{
	public static void main(String[] args)
	{
		Toolkit kit = Toolkit.getDefaultToolkit();
		Dimension screenSize = kit.getScreenSize();
		int screenWidth = screenSize.width;
		int screenHeight = screenSize.height;
		JFrame frame = new JFrame("My frame");
		frame.setSize(screenWidth,screenHeight);
		frame.setResizable(false);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.setVisible(true);
	}
}
```

Để người dùng không thay đổi được size của mình, dùng frame.setResizable(false)

Bài 5 - Các đối tượng đồ họa cơ bản của Java

* Button
Button button=new Button("OK");
add(button);
hoặc add(new Button("OK"));
Button sử dụng ActionListener để nghe sự kiện và truyền hành động
* Label
Label label=new Label("The sum of values here:");
Label là nhãn
* Panel
Panel panel=new Panel();
Panel là khung chứa.

Bài 6 – Checkbox

Checkbox dùng để chuyển đổi trạng thái (state) giữa yes/no hay true/false. Khi state là true thì ô đã được đánh dấu. Có 3 instructor thường dùng là:
Checkbox() Checkbox(String label) Checkbox(String label,boolean state) với label hiển thị nhãn còn state là true/false
Để xác lập state cho một Checkbox ta dùng phương thức setState(true)
Để lấy state hiện hành của một Checkbox ta dùng phương thức getState()
Để xử lí tình huống của Checkbox khi nó thay đổi trạng thái, ta phải cho nó implements giao diện ItemListener, và bên trong nó có phương thức itemStateChanged(ItemEvent

e). Còn để Checkbox thực hiện những hành động của lớp ấy thì ta phải dùng phương thức addItemListener.

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {
                JFrame frame = new JFrame("My frame");
                Checkbox checkbox=new Checkbox("Documents",false);
                checkbox.addItemListener(new ItemListener()
                {
                        public void itemStateChanged(ItemEvent e)
                        {
                                System.out.println("Changed");
                        }
                });
                frame.add(checkbox);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.pack();
                frame.setVisible(true);
        }
}
```

Ở ví dụ này thì mỗi lần bạn thay đổi trạng thái Checkbox, màn hình Console sẽ in ra câu "Changed" Bây giờ nếu bạn muốn màn hình in ra chỉ khi nào nó được chọn mà thôi, thì sửa lại phương thức itemStateChanged như sau
if(e.getStateChange()==ItemEvent.SELECTED) System.out.println("Changed");
SELECTED và DESELECTED là 2 hằng số biển diễn trạng thái true hay false của Checkbox

Bài 7 - Checkbox nhiều tùy chọn (CheckboxGroup)

Đầu tiên, hãy tạo một nhóm Checkbox như sau CheckboxGroup g=new CheckboxGroup();
Sau đó đưa các Checkbox muốn đưa vào nhóm Checkbox đó như sau
Checkbox c1=new Checkbox("Option 1",g,true);
Checkbox c2=new Checkbox("Option 2",g,false);
Checkbox c2=new Checkbox("Option 2",g,false);
Cả 3 cái cùng mang giá trị false cũng được, nhưng nếu là true thì chỉ được một cái true
Bài tập sau sẽ tạo một CheckboxGroup có 3 Checkbox. Để listener biết là Checkbox nào được chọn, ta dùng phương thức getItem (trả về Object) Lưu ý là để cả 3 Checkbox cùng hiển thị trên frame, ta dùng Panel

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
```

```
{
        public static void main(String args[])
        {
                JFrame frame = new JFrame("My frame");
                CheckboxGroup g=new CheckboxGroup();
                Checkbox c1=new Checkbox("Option 1",g,true);
                Checkbox c2=new Checkbox("Option 2",g,false);
                Checkbox c3=new Checkbox("Option 3",g,false);
                MyItemListener listener = new MyItemListener();
                c1.addItemListener(listener);
                c2.addItemListener(listener);
                c3.addItemListener(listener);
                Panel panel=new Panel();
                frame.add(panel);
                panel.add(c1);
                panel.add(c2);
                panel.add(c3);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.pack();
                frame.setVisible(true);
        }
}
class MyItemListener implements ItemListener
{
        public void itemStateChanged(ItemEvent e)
        {
                if(e.getStateChange()==ItemEvent.SELECTED)
                {
                        Object temp=e.getItem();
                        String s=(String)temp;
                        System.out.println(s);
                }
        }
}
```

Lưu ý (String)temp thực ra là lấy cái label của Object temp. 2 lệnh có thể thay thế bằng 1 lệnh String s=(String)e.getItem();

Bài 8 – Choice

Choice myChoice = new Choice();
sau đó đưa mục chọn vào Choice như sau
myChoice.addItem("Red");
myChoice.addItem("Green");
myChoice.addItem("Blue");
Khi đó 3 mục chọn được đánh số lần lượt là 0,1,2 (đặt là i: thứ tự mục chọn)
Để bỏ mục chọn nào ra khỏi Choice, ta dùng myChoice.remove(i) với i là thứ tự mục chọn
Để bỏ tất cả mục chọn khỏi Choice, ta dùng myChoice.removeAll()
Để chọn mục chọn nào trong Choice, ta dùng muChoice.select(i)

Lưu ý là ta có thể dùng số thứ tự hoặc nhãn đều được, ví dụ myChoice.remove("Blue") hay myChoice.remove(2) đều được. Và nếu có 10 mục chọn có nhãn là "Blue" thì myChoice.remove("Blue") chỉ xóa mục chọn đầu tiên nó tìm thấy

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {
                JFrame frame = new JFrame("My frame");
                Choice myChoice = new Choice();
                myChoice.addItem("Red");
                myChoice.addItem("Green");
                myChoice.addItem("Blue");
                myChoice.addItemListener(new ItemListener()
                {
                        public void itemStateChanged(ItemEvent e)
                        {
                                if(e.getStateChange()==ItemEvent.SELECTED)
                                {
                                        String s=(String)e.getItem();
                                        System.out.println(s);
                                }
                        }
                });
                frame.add(myChoice);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.pack();
                frame.setVisible(true);
        }
}
```

Bài 9 – List

Với Checkbox, ta chỉ có thể chọn giữa 2 trạng thái true/false của một đối tượng. Với CheckboxGroup và Choice, ta chỉ có thể một trong các đối tượng. Với List, ta có thể chọn một vài đối tượng, thậm chí chọn hết.
Mặc định của một List là chỉ hiển thị tối đa 4 phần tử. Các phương thức khởi tạo:
List() sẽ tạo một List mỗi lần chọn chỉ chọn được một hàng (đơn chọn)
List(int num) sẽ tạo một danh sách mỗi lần chọn mỗi lần chọn chỉ chọn được một hàng, nhưng sẽ hiển thị num hàng chứ không phải là 4 như mặc định
List(int num,boolean multiMode) y chang cái trên, nhưng thêm là mỗi lần chọn được chọn nhiều phần tử một lúc (nếu multiMode là true) (đa chọn)
Như vậy List(7) và List(7,false) là như nhau, hiển thị 7 hàng một lúc và mỗi lần chọn chỉ chọn được một hàng
Để add phần tử vào List:

List myList=new List(3,true);
myList.add("Pascal");
myList.add("C\\C++");
myList.add("VB");
myList.add("Java");
Các phần tử cũng được đánh thứ tự từ 0. Để thêm phần tử vào vị trí nào ta đưa vào chỉ số ta thích, ví dụ
myList.add("Assembler",0);
Để thay thế một phần tử tại vị trí nào ta dùng phương thức
myList.replaceItem("VB.NET",2); //VB bị thay bằng VB.NET
Để xóa một phần tử nào ta dùng phương thức remove(i)
myList.remove(3); hay myList.remove("Java"); đều được. Và nếu có 10 mục chọn có nhãn là "Java" thì myList.remove("Java") chỉ xóa phần tử đầu tiên nó tìm thấy
Để xóa tất cả ta dùng myList.removeAll();
Để chọn phần tử và bỏ chọn phần tử ta dùng select(i) và deselect(i)

Bài 10 - Làm việc với List

* Với List đơn chọn
Để biết được phần tử nào đã được chọn, ta dùng 2 phương thức int getSelectedIndex() và String getSelectedItem()
int getSelectedIndex() sẽ trả về số thứ tự của phần tử đã được chọn, nếu không có phần tử nào thì trả về -1
String getSelectedItem() sẽ trả về label của phần tử đã được chọn, nếu không có phần tử nào thì trả về ""

```
final List l=new List();
l.add("Pascal");
l.add("C\\C++");
l.add("VB");
l.add("Java");
frame.add(l);
l.addItemListener(new ItemListener()
{
        public void itemStateChanged(ItemEvent e)
        {
                System.out.println(l.getSelectedIndex()+"
"+l.getSelectedItem());
        }
});
```

* Với List đa chọn
Để biết được những phần tử nào đã được chọn, ta dùng 2 phương thức int[] getSelectedIndexs() và String[] getSelectedItems()
int[] getSelectedIndexs() là một mảng sẽ trả về những số thứ tự của các phần tử đã được chọn, nếu không có phần tử nào thì trả về -1

String[] getSelectedItems() là một mảng sẽ trả về những label của các phần tử đã được chọn, nếu không có phần tử nào thì trả về ""

```
final List l=new List(3,true);
l.add("Pascal");
l.add("C\\C++");
l.add("VB");
l.add("Java");
frame.add(l);
l.addItemListener(new ItemListener()
{
        public void itemStateChanged(ItemEvent e)
        {
                int[] a=l.getSelectedIndexs();
                String[] b=l.getSelectedItems();
                for(int i=0;i<a.length;i++)
                        System.out.println(a|i|);
                for(int j=0;j<b.length;j++)
                        System.out.println(b[j]);
        }
});
```

Bài 11 - TextField và TextArea

* TextField
Có 4 phương thức khởi tạo
TextField() không thêm gì cả
TextField(String s) với s là chuỗi ban đầu xuất hiện trên TextField (ví dụ "Input your name here")
TextField(int num) với num là độ dài TextField
TextField(String s,int num) với s là chuỗi ban đầu xuất hiện trên TextField và num là độ dài TextField
* TextArea
Có 4 phương thức khởi tạo
TextArea() không thêm gì cả
TextArea(String s) với s là chuỗi ban đầu xuất hiện trên TextArea
TextArea(int row,int column) với row và column là số hàng và cột trên TextField
TextField(String s,int row,int column) là đầy đủ nhất kết hợp 2 phương thức khởi tạo trên
* Các phương thức của TextField và TextArea
Đưa nội dung văn bản vào bằng void setText(String txt)
Lấy nội dung văn bản ra bằng String getText()
Lấy nội dung văn bản đang được đánh dấu (bôi đen) ra bằng String getSelectedText()
Để người dùng không thể thay đổi nội dung bằng void setEditable(false) (mặc định là true)
Đặc biệt là TextField có một phương thức mà TextArea không có, đó là void setEchoChar(char c). Ví dụ setEchoChar('*') thì phương thức này sẽ khiến cho tất cả kí tự nhập vào TextField đều chỉ hiển thị là kí tự '*' (rất hữu dụng khi nhập password)
* Phương thức dùng chung với TextField

Giả sử nếu bạn muốn làm ra một chương trình bảng tính, dữ liệu nhập vào TextField, bây giờ muốn chuyển dữ liệu ấy ra số thực để tính toán
s=textField.getText();
value1=Float.parseFloat(s);
Tương tự với Byte.parseByte,Integer.parseInt,Double.parseDouble,...

Bài 12 - Scrollbar (thanh trượt)

Scrollbar được cập nhật theo 3 tình huống unit,block và absolute
- Khi người dùng click chuột vào mũi tên ở 2 đầu Scrollbar thì unit nảy sinh, thanh trượt sẽ tự động trừ đi hay cộng thêm vị trí của con trượt 1 đơn vị (ta có thể thay đổi giá trị này, mặc định là 1)
- Khi người dùng click chuột vào khoảng giữa thanh trượt và vị trí hiện hành thì block nảy sinh, con trượt sẽ dịch chuyển một khoảng là block
- Khi người dùng nắm vào vị trí hiện tại của con trượt và lôi (drag) nó từ vị trí này sang vị trí khác, absolute nảy sinh
* Khởi tạo thanh trượt
Scrollbar() là đơn giản nhất, mặc định là thanh trượt đứng
Scrollbar(int orientation) với orientation là Scrollbar.HORIZONTAL (ngang) hay Scrollbar.VERTICAL (đứng)
Scrollbar(int orientation,int position,int block,int min,int max) là đầy đủ nhất, ví dụ Scrollbar(Scrollbar.HORIZONTAL,50,15,0,100) tức là thanh trượt ngang, phạm vi từ 0 đến 100, vị trí ban đầu của con trượt là 50 (giữa thanh) khi tình thuống block xảy ra thì con trượt di chuyển 15
* Các phương thức của thanh trượt
Để thay đổi giá trị unit (mặc định là 1) ta dùng setUnitIncrement(int unit) với unit mới
Để thay đổi giá trị block ta dùng setBlockIncrement(int block) với block mới
Để biết vị trí hiện hành của con trượt ta dùng int getValue()
* Viết bộ nghe và hành động cho thanh trượt
Để xử lí tình huống của Scrollbar khi nó thay đổi trạng thái, ta phải cho nó implements giao diện AdjustmentListener, và bên trong nó có phương thức adjustmentValueChanged(AdjustmentEvent e). Còn để Scrollbar thực hiện những hành động của lớp ấy thì ta phải dùng phương thức addAdjustmentListener.
Tuy vậy, một thanh trượt thì mỗi khi tác động đến nó, nó phải "cuộn" một cái gì đấy. Muốn AdjustmentListener đáp ứng mỗi khi ta "cuộn" thì cần AdjustmentEvent e biết được vị trí con trượt đang ở đâu, ta dùng e.getValue(). Ví dụ sau là thanh trượt và TextField

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {
```

```
                JFrame frame = new JFrame("My frame");
                Panel p=new Panel();
                frame.add(p);
                Scrollbar s=new Scrollbar(Scrollbar.HORIZONTAL,0,10,0,100);
                p.add(s);
                final TextField t=new TextField(100);
                t.setEditable(false);
                p.add(t);
                s.addAdjustmentListener(new AdjustmentListener()
                {
                        public void adjustmentValueChanged(AdjustmentEvent e)
                        {
                                int currentPos=e.getValue();
                                String text="";
                                for(int i=0;i<currentPos;i++) text+='*';
                                t.setText(text);
                        }
                });
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.pack();
                frame.setVisible(true);
        }
}
```

Bây giờ thì mỗi lần bạn kéo con trượt thì kí tự '*' lạy thay đổi trong TextField t.

Bài 13 - Các đối tượng khung chứa (container) và bộ quản lí trình bày (layout manager)

Khung chứa nói nôm na là những cái khung tranh bạn cần để dán những nét vẽ (đối tượng đồ họa) của bạn lên đó. Và để sắp xếp các đối tượng này trên khung chứa thì ca cần tới bộ quản lí trình bày.
- Khung chứa mà bạn đã quen thuộc là Frame. Nó giống như cửa sổ của Windows và chứa toàn bộ ứng dụng của bạn
- Một khung chứa khác mà bạn đã biết qua là Panel. Nó giống như một cái bảng hiển thi đầy đủ các thành phần GUI mà bạn muốn tống lên Frame
- Những cái còn lại sẽ học sau. Tất cả các lớp Frame,Panel,... đều là con của lớp Container
* Các phương thức của Container (các phương thức chung của các đối tượng khung chứa)
Component add(Component c) đưa một đối tượng c vào khung chứa. Ví dụ frame.add(panel);
void remove(Component c) đưa một đối tượng c ra khỏi khung chứa. Ví dụ frame.remove(panel);

Bài 14 - Nhắc lại về Jframe

Phần 1 đã nói về JFrame, bây giờ chỉ nhắc lại

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
	public static void main(String args[])
	{
		//frame khong phu thuoc do phan giai man hinh
		Toolkit kit = Toolkit.getDefaultToolkit();
		Dimension screenSize = kit.getScreenSize();
		int screenWidth = screenSize.width;
		int screenHeight = screenSize.height;
		int frameWidth = 200;
		int frameHeight = 60;
		JFrame frame = new JFrame("My frame");
		frame.setLayout(null);
		frame.setLocation((screenWidth-frameWidth)/2,(screenHeight-
frameHeight)/2);
		frame.setSize(frameWidth,frameHeight);
		frame.setResizable(false);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

		//tao Button Exit cai dat phuong thuc hanh dong qua class
rieng
		Button b1 = new Button("Exit");
		b1.setBounds(0,0,100,25);
		frame.add(b1);
		EventQuit eventQuit=new EventQuit();
		b1.addActionListener(eventQuit);

		//tao Button About cai dat phuong thuc hanh dong truc tiep
		final Button b2 = new Button("About");
		b2.setBounds(100,0,100,25);
		frame.add(b2);
		b2.addActionListener(new ActionListener()
		{
			public void actionPerformed(ActionEvent e)
			{
				if(e.getSource()==b2) System.out.println("Made
in Vietnam");
			}
		});
		frame.setVisible(true);
	}
}
class EventQuit implements ActionListener
{
	public void actionPerformed(ActionEvent e)
	{
		System.exit(0);
	}
}
```

Bài trên đã dùng lại các kiến thức đã học trước đây: setLayout(null) và setBounds, đặt ứng dụng giữa màn hình và không phụ thuộc phân giải màn hình nhờ dùng Toolkit, 2 cách cài đặt phương thức hành động qua class riêng và cài đặt trực tiếp.

Bài 15 - Tạo và add hàng loạt button

Bạn hãy cứ tưởng tượng nếu bạn phải add khoảng 30 button vào Frame của mình, bạn phải viết khoảng 30 câu lệnh khởi tạo, add rất là mệt. Hãy để máy tự động làm cho bạn, chỉ với vài vòng lặp. Bạn cần dùng một bảng String để lưu những label của button và một mảng Button để lưu chính những button. Đồng thời cũng dựa vào mảng Button đấy để cài đặt phương thức hành động

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core extends JFrame implements ActionListener
{
        Panel p=new Panel();
        final String[]
a={"File","Edit","View","Insert","Format","Table","Windows","Help"};
        final Button[] b=new Button[a.length];
        public Core(String title)
        {
                setTitle(title);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                add(p);
                setSize(500,60);
                for(int i=0;i<a.length;i++)
                {
                        b|i|=new Button(a|i|);
                        p.add(b|i|);
                        b|i|.addActionListener(this);
                }
                setVisible(true);
        }
        public void actionPerformed(ActionEvent e)
        {
                for(int i=0;i<a.length;i++)
                        if(e.getSource()==b|i|) System.out.println("You have
clicked button "+a|i|);
        }
        public static void main(String args[])
        {
                Core c=new Core("My frame");
        }
}
```

Sở dĩ trong bài này ta dùng khởi tạo của Core là để sử dụng addActionListener(this);

Bài 16 - MenuBar,Menu và MenuItem

* MenuBar và Menu

Để thấy được các Menu như File, Edit, Help như trên một cửa sổ Windows thông thường thì tất cả các đối tượng Menu ấy phải được add vào một MenuBar. Để Menubar có thể xúât hiện trong JFrame thì ta dùng phương thức setMenuBar(menuBar). Chương trình sau minh họa một ứng dụng như vậy

```
import javax.swing.JFrame;
import java.awt.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                MenuBar menuBar=new MenuBar();
                f.setMenuBar(menuBar);
                Menu file=new Menu("File");
                menuBar.add(file);
                Menu edit=new Menu("Edit");
                menuBar.add(edit);
                Menu help=new Menu("Help");
                menuBar.setHelpMenu(help);
                //phan code duoi them vao day
                f.setSize(200,60);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

* Menu và MenuItem

Còn để New, Open, Save hiện ra trong menu File thì các MenuItem ấy phải được add vào menu File. Ta thêm vào như sau

```
MenuItem newItem=new MenuItem("New");
file.add(newItem);
MenuItem openItem=new MenuItem("Open");
file.add(openItem);
MenuItem saveItem=new MenuItem("Save");
file.add(saveItem);
saveItem.setEnable(false);
//phan code duoi them vao day
file.addSeparator(); //phuong thuc nay dua mot hang phan cach vao menu File
MenuItem exitItem=new MenuItem("Exit");
file.add(exitItem);
```

Để cho một MenuItem không thể chọn được, ta dùng phương thức setEnable(false) (mặc định là true) ví dụ như trên saveItem.setEnable(false); Điều này đặc biệt hữu ích với ứng dụng văn bản chưa có chữ nào thì không nên cho người dùng chọn MenuItem

saveItem
* Menu và submenu
Để tạo một MenuItem chứa một Menu khác (submenu), ta chỉ việc tạo Menu đó rồi add vào menu item kia là xong. Ta thêm vào như sau
Menu print=new Menu("Setup Print");
file.add(print);
MenuItem previewItem=new MenuItem("Preview");
print.add(previewItem);
MenuItem printItem=new MenuItem("Print");
print.add(printItem);
//phan code duoi them vao day
* CheckboxMenuItem
Bạn cũng có thể tạo một mục chọn có khả năng đánh dấu bằng cách sử dụng lớp CheckboxMenuItem
CheckboxMenuItem autosave=new CheckboxMenuItem("Auto Save");
file.add(autosave);
Ngoài ra còn một phương thức khởi tạo khác là CheckboxMenuItem autosave=new CheckboxMenuItem("Auto Save",true); Mặc định là false (chưa chọn)

Bài 17 - Khởi tạo và cài đặt phương thức hành động hàng loạt

Bạn hãy cứ tưởng tượng nếu bạn phải làm một ứng dụng giống như Microsoft Word (chỉ mới nói Word thôi chứ chưa dám đụng tới Photoshop, Corel gì cả) với một lô Menu và mỗi Menu có hơn chục cái MenuItem. Vậy thì bạn phải khởi tạo, phải add rất là mệt. Hãy để máy tự động làm cho bạn, chỉ với vài vòng lặp. Bạn cần dùng một bảng String để lưu những label của menu và một mảng Menu để lưu chính những menu.

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                MenuBar menuBar=new MenuBar();
                f.setMenuBar(menuBar);
                final String[]
menuLabel={"File","Edit","View","Insert","Format","Table","Windows","Help"};
                final Menu[] menu=new Menu[menuLabel.length];
                for(int i=0;i<menuLabel.length;i++)
                {
                        menu|i|=new Menu(menuLabel|i|);
                        menuBar.add(menu|i|);
```

```
                }
                final String[] fileMenuItemLabel={"New","Open","Save","Exit"};
                final MenuItem[] fileMenuItem=new
MenuItem[fileMenuItemLabel.length];
                for(int i=0;i<fileMenuItemLabel.length;i++)
                {
                        fileMenuItem|i|=new MenuItem(fileMenuItemLabel|i|);
                        menu[0].add(fileMenuItem|i|);
                        if(i==2) menu[0].addSeparator();
                }
                fileMenuItem[3].addActionListener(new ActionListener()
                {
                        public void actionPerformed(ActionEvent e)
                        {
                                System.exit(0);
                        }
                });
                f.setSize(400,60);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

Bài 18 – ScrollPane

```
import javax.swing.JFrame;
import java.awt.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                ScrollPane s=new ScrollPane();
                TextArea t=new TextArea();
                s.add(t);
                f.add(s);
                f.setSize(200,120);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

Bài 19 - LookAndFeel (cảm quan giao diện)

LookAndFeel (viết tắt là LaF gọi là cảm quan giao diện cho ứng dụng. Sử dụng rất đơn giản
UIManager.setLookAndFeel(String className) với UIManager là một class còn className là tên class chứa cái LaF đó. Java hỗ trợ sẵn 3 cái là:
javax.swing.plaf.metal.MetalLookAndFeel (giao diện Java)
com.sun.java.swing.plaf.windows.WindowsLookAndFeel (giao diện Windows)

com.sun.java.swing.plaf.motif.MotifLookAndFeel (giao diện UNIX)
Sau khi set, để giao diện hiển thị trên JFrame nào, cần cập nhật trên JFrame đó bằng phương thức sau
SwingUtilities.updateComponentTreeUI(myFrame) (myFrame là tên JFrame cần cập nhật)
Ví dụ sau sẽ minh họa cách thay đổi LaF dựa vào CheckboxGroup. Cần nói thêm là phương thức UIManager.setLookAndFeel(String className) bắt buộc phải xử lí ngoại lệ

```
import javax.swing.JFrame;
import java.awt.*;
import java.awt.event.*;
class Core
{
	public static void main(String args[])
	{
		final JFrame f = new JFrame("My JFrame");
		final String[] a={"Metal","Windows","Motif"};
		final Checkbox[] b=new Checkbox[a.length];
		final String[] c=new String[a.length];
		CheckboxGroup g=new CheckboxGroup();
		c[0]="javax.swing.plaf.metal.MetalLookAndFeel";
		c[1]="com.sun.java.swing.plaf.windows.WindowsLookAndFeel";
		c[2]="com.sun.java.swing.plaf.motif.MotifLookAndFeel";
		Panel p=new Panel();
		f.add(p);
		for(int i=0;i<a.length;i++)
		{
			b|i|=new Checkbox(a|i|,g,false);
			p.add(b|i|);
			b|i|.addItemListener(new ItemListener()
			{
				public void itemStateChanged(ItemEvent e)
				{
					for(int j=0;j<a.length;j++)
					{
						if(e.getSource()==b[j])
						{
							try
							{

	UIManager.setLookAndFeel(c[j]);
							}
							catch(Exception
exception)
							{

	System.out.println("LaF not found");
							}

	SwingUtilities.updateComponentTreeUI(f);
						}
					}
				}
			});
		}
		f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
                f.setVisible(true);
        }
}
```

Bài 20 - Cách sử dụng LaF của hãng thứ 3:

Cách 1: coi nó như là 1 add-in plugin, tức là đưa cái file .jar chứa class mình cần vào thư mục jdk1.5.0\jre\lib\ext rồi cứ thế điều chỉnh className cho phù hợp

Cách 2: giải nén file .jar ra và tống thẳng nó vào gói jdk1.5.0\jre\lib\rt.jar
Ví dụ: bạn DOWN được gói xplookandfeel.jar và chép nó theo cách 1
Bạn hãy mở file readme của gói này ra và tìm thấy className của nó là "com.stefankrause.xplookandfeel.XPLookAndFeel" thì đưa thêm nó vào
* LaF có bản quyền (ở đây lấy ví dụ là Alloy-có cả mã)
Để sử dụng LaF có bản quyền bạn setProperty cho nó, ví dụ là dùng AlloyLaF (className là "com.incors.plaf.alloy.AlloyLookAndFeel")

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {

        com.incors.plaf.alloy.AlloyLookAndFeel.setProperty("alloy.licenseCode"
,"v#ej_technologies#uwbjzx#e6pck8");
                final JFrame f = new JFrame("My f");
                final String[] a={"Metal","Windows","Motif","XP","Alloy"};
                final Checkbox[] b=new Checkbox[a.length];
                final String[] c=new String[a.length];
                CheckboxGroup g=new CheckboxGroup();
                c[0]="javax.swing.plaf.metal.MetalLookAndFeel";
                c[1]="com.sun.java.swing.plaf.windows.WindowsLookAndFeel";
                c[2]="com.sun.java.swing.plaf.motif.MotifLookAndFeel";
                c[3]="com.stefankrause.xplookandfeel.XPLookAndFeel";
                c[4]="com.incors.plaf.alloy.AlloyLookAndFeel";
                Panel p=new Panel();
                f.add(p);
                for(int i=0;i<a.length;i++)
                {
                        b|i|=new Checkbox(a|i|,g,false);
                        p.add(b|i|);
                        b|i|.addItemListener(new ItemListener()
                        {
                                public void itemStateChanged(ItemEvent e)
                                {
                                        for(int j=0;j<a.length;j++)
                                        {
                                                if(e.getSource()==b[j])
                                                {
                                                        try
                                                        {
```

```
        UIManager.setLookAndFeel(c[j]);
                                                        }
                                                        catch(Exception
exception)
                                                        {

        System.out.println("LaF not found");
                                                        }

        SwingUtilities.updateComponentTreeUI(f);
                                                    }
                                                }
                                            }
                                    });
                }
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

Bài 21 – Jcheckbox

* JCheckBox tương tự Checkbox

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class Core
{
        public static void main(String args[])
        {
                JFrame frame = new JFrame("My frame");
                JCheckbox checkbox=new JCheckbox("Documents",false);
                checkbox.addItemListener(new ItemListener()
                {
                        public void itemStateChanged(ItemEvent e)
                        {
                                System.out.println("Changed");
                        }
                });
                frame.add(checkbox);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setVisible(true);
        }
}
```

Bài 22 - JRadioButton và ButtonGroup

* JRadioButton và ButtonGroup tương tự Checkbox và CheckboxGroup

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class Core
```

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

        com.incors.plaf.alloy.AlloyLookAndFeel.setProperty("alloy.licenseCode"
,"v#ej_technologies#uwbjzx#e6pck8");
                final JFrame f = new JFrame("My f");
                final String[] a={"Metal","Windows","Motif","XP","Alloy"};
                final JRadioButton[] b=new JRadioButton[a.length];
                final String[] c=new String[a.length];
                c[0]="javax.swing.plaf.metal.MetalLookAndFeel";
                c[1]="com.sun.java.swing.plaf.windows.WindowsLookAndFeel";
                c[2]="com.sun.java.swing.plaf.motif.MotifLookAndFeel";
                c[3]="com.stefankrause.xplookandfeel.XPLookAndFeel";
                c[4]="com.incors.plaf.alloy.AlloyLookAndFeel";
                ButtonGroup g=new ButtonGroup();
                JPanel p=new JPanel();
                f.add(p);
                for(int i=0;i<a.length;i++)
                {
                        b|i|=new JRadioButton(a|i|);
                        g.add(b|i|);
                        p.add(b|i|);
                        b|i|.addItemListener(new ItemListener()
                        {
                                public void itemStateChanged(ItemEvent e)
                                {
                                        for(int j=0;j<a.length;j++)
                                        {
                                                if(e.getSource()==b[j])
                                                {
                                                        try
                                                        {

        UIManager.setLookAndFeel(c[j]);
                                                        }
                                                        catch(Exception
exception)
                                                        {

        System.out.println("LaF not found");
                                                        }

        SwingUtilities.updateComponentTreeUI(f);
                                                }
                                        }
                                }
                        });
                }
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

Bài 23 - JComboBox và Jlist

* JComboBox tương tự như Choice

```
import javax.swing.*;
import java.awt.*;
class Core
{
	public static void main(String args[])
	{
		JFrame f = new JFrame("My frame");
		String[] label={"ASM","C\\C++","VB","Java"};
		JComboBox box=new JComboBox(label);
		f.add(box);
		f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		f.setSize(100,55);
		f.setVisible(true);
	}
}
```

* JList tương tự List nhưng nó lại không tự kéo thả được như List, cần có sự hỗ trợ của JScrollPane

```
import javax.swing.*;
import java.awt.*;
class Core
{
	public static void main(String args[])
	{
		JFrame f = new JFrame("My frame");
		String[] label={"ASM","Pascal","C\\C++","VB","Java"};
		JList l=new JList(label);
		ScrollPane s=new ScrollPane();
		s.add(l);
		f.add(s);
		f.setSize(100,100);
		f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		f.setVisible(true);
	}
}
```

* Còn lại thì JMenuBar,JMenu,JMenuItem tương tự MenuBar,Menu,MenuItem

Bài 24 – JtabbedPane

Đây gọi là đối tượng phân trang. Ví dụ dưới đây minh họa 1 trong các phương thức addTab là JTabbedPane.addTab(String title,Component component) Các component trong ví dụ đều là các JButton

```
import javax.swing.*;
import java.awt.*;
class Core
{
	public static void main(String args[])
```

```
        {
                JFrame f = new JFrame("My frame");
                String[] label={"ASM","Pascal","C\\C++","VB","Java"};
                JButton[] b=new JButton[label.length];
                JTabbedPane p=new JTabbedPane();
                for(int i=0;i<label.length;i++)
                {
                        b|i|=new JButton(label|i|);
                        p.addTab(label|i|,b|i|);
                }
                f.add(p);
                f.setSize(300,100);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

Thông thường thì các componenet đều là các khung chứa (ví dụ như JPanel) chứa nhiều đối tượng riêng biệt

Bài 25 – JtoolBar

JToolBar (thanh công cụ) giống y như bạn thường thấy trong các ứng dụng Windows. Nó bao gồm nhiều JButton, mỗi JButton có một Icon riêng. Trước hết hãy chuẩn bị vài hình .gif 24x24 cho ví dụ này. Nếu không tìm thấy thì chép từ jdk1.5.0\demo\plugin\jfc\Stylepad\resources. Chép vào thư mục chứa mã nguồn

```
import javax.swing.*;
import java.awt.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                JToolBar t=new JToolBar();
                String[]
label={"New","Open","Save","Cut","Copy","Paste","Bold","Italic","Underline","
Left","Right","Center"};
                String[]
file={"new.gif","open.gif","save.gif","cut.gif","copy.gif","paste.gif","bold.
gif","italic.gif","underline.gif","left.gif","right.gif","center.gif"};
                ImageIcon[] icon=new ImageIcon[label.length];
                JButton[] b=new JButton[label.length];
                for(int i=0;i<label.length;i++)
                {
                        icon|i|=new ImageIcon(file|i|);
                        b|i|=new JButton(icon|i|);
                        b|i|.setToolTipText(label|i|);
                        t.add(b|i|);
                        if((i==2)||(i==5)||(i==8)) t.addSeparator();
                }
                f.add(t);
                f.setSize(600,70);
```

```
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

Bài 26 – Jtable

* JTable hiển thị dữ liệu có sẵn

```
import javax.swing.*;
import java.awt.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                String[][] dat={{"1","JCreator","Xinox","Beginer"},

        {"2","jGRASP","Auburn","Medium"},
                                                {"3","NetBeans","Sun
Microsystems","Expert"},

        {"4","Gel","GExperts","Beginer"},

        {"5","Eclipse","Eclipse","Expert"},

        {"6","JBuilder","Borland","Expert"}};
                String[] columnName={"ID","Name","Company","Rank"};
                JTable t=new JTable(dat,columnName);
                JScrollPane s=new JScrollPane(t);
                f.add(s);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setSize(400,100);
                f.setVisible(true);
        }
}
```

* Thao tác trực tiếp dữ liệu trong từng ô của bảng

Trong ví dụ dưới chúng ta sẽ thay đổi dữ liệu của cột "Rank" bằng một JComboBox, sử dụng class TableColumn

```
import javax.swing.*;
import java.awt.*;
import javax.swing.table.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                String[][] dat={{"1","JCreator","Xinox","Beginer"},

        {"2","jGRASP","Auburn","Medium"},
                                                {"3","NetBeans","Sun
Microsystems","Expert"},

        {"4","Gel","GExperts","Beginer"},
```

```
        {"5","Eclipse","Eclipse","Expert"},

        {"6","JBuilder","Borland","Expert"}};
                String[] columnName={"ID","Name","Company","Rank"};
                JTable t=new JTable(dat,columnName);
                JScrollPane s=new JScrollPane(t);
                f.add(s);
                JComboBox c=new JComboBox(new
String[]{"Low","High","Extremely"});
                TableColumn rankColumn=t.getColumn("Rank");
                rankColumn.setCellEditor(new DefaultCellEditor(c));
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setSize(400,100);
                f.setVisible(true);
        }
}
```

Bài 27 - JOptionPane cơ bản

Đây có thể nói là công cụ Dialog mạnh nhất
Trước tiên hãy xem qua ví dụ sau

```
import javax.swing.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                JOptionPane.showMessageDialog(f,"Hien thi cau thong bao","Hien
thi tieu de",JOptionPane.ERROR_MESSAGE);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

JOptionPane bao gồm các thành phần chính sau đây: Title, Icon, Message, InputValue và OptionButtons. Không cần bao gồm đủ tất cả
* Khởi tạo của JOptionPane
JOptionPane(Object message, int messageType, int optionType, Icon icon, Object[] options)
Bạn có thể thiếu bất cứ thành phần nào thậm chí có thể thiếu hết
Object message: câu thông báo hiển thị trên JOptionpane
int messageType: bao gồm ERROR_MESSAGE, INFORMATION_MESSAGE, WARNING_MESSAGE, QUESTION_MESSAGE, và PLAIN_MESSAGE
int optionType: bao gồm DEFAULT_OPTION, YES_NO_OPTION, YES_NO_CANCEL_OPTION, OK_CANCEL_OPTION
Icon icon: hình icon của JOptionPane

Ví dụ:

```
import javax.swing.*;
class Core
{
        public static void main(String args[])
        {
                JFrame f = new JFrame("My frame");
                f.add(new JOptionPane("Hien thi cau thong bao",
JOptionPane.QUESTION_MESSAGE, JOptionPane.YES_NO_CANCEL_OPTION));
                f.setSize(250,150);
                f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                f.setVisible(true);
        }
}
```

* Các phương thức (hay sử dụng hơn)

void showMessageDialog(Component parentComponent, Object message, String title, int messageType)

String showInputDialog(Component parentComponent, Object message, String title, int messageType)

int showConfirmDialog(Component parentComponent, Object message, String title, int optionType, int messageType) Yes trả về 0 và No trả về 1