**ÔN THI PE PRO192**

1. **For**

* Vòng lặp for trong Java được sử dụng để lặp qua một dải giá trị. Chúng ta có thể sử dụng vòng lặp for để lặp qua một mảng, một danh sách, java set…
* Cấu trúc:

For(Giá trị đầu; giá trị cuối;bước nhảy)

{

…..

}

* Example:

For (int i=1; i<=7; i=i+2)

{

System.out.println("FOR LOOP " + i);

}

1. **Hàm**

* Chúng ta sẽ có **2 loại hàm**, mỗi loại sẽ có cách gọi khác nhau:
* Hàm có trả về kết quả
* Hàm không trả về kết quả

### Hàm có trả về kết quả

* Đối với hàm có kết quả trả về, chúng ta cần dùng từ khoá return để trả về kết quả mà nó đã tính toán được.
* Example

public class Main {

public static int sum(int a, int b) {

return a + b;

}

public static void main(String[] args) {

int sum = sum(2,5);

System.out.println(sum);

}

}

### Hàm không trả về kết quả

* Đối với loại hàm này chúng ta chỉ cần gọi để sử dụng.
* Example

public class Main {

public static void sum(int a, int b)

{

System.out.println(a+b);

}

public static void main(String[] args)

{

sum(2,5);

}

}

* Kiểu boolean dùng trong hàm: trả về true/false

1. **String**

* split(“...”): Cắt chuỗi tại vị trí…

String s = "share programming net";

String[] splits = s.split(" ");

for (int i=0;i<splits.length;i++)

System.out.println(splits[i]);

* toUpperCase:Viết hoa
* toLowerCase:Viết thường
* substring(a,b): lấy string trong đoạn a đến b

1. **Getter,setter**

* Phương thức get trả về giá trị biến
* Phương thức set đặt giá trị.
* Example

public class Person {

private String name;

// Getter

public String getName() {

return name;

}

// Setter

public void setName(String newName) {

this.name = newName;

}

}

VD: t.setR(5): thay giá trị của R bằng 5

VD: t.getR(): lấy giá trị của R

1. **Interface - implements**
2. **Abstract - extends**
3. **Maths**

Cú pháp :Math.tenphuongthuc

Làm tròn k đến số thập phân thứ n: System.out.printf("%.nf",k);

| Tên phương thức | Mô tả |
| --- | --- |
| PI | Trả về giá trị PI |
| abs(a) | Trả về giá trị tuyệt đối của a |
| sqrt(a) | Trả về căn bậc 2 của số a |
| pow(x,y) | Trả về x mũ y |
| ceil(a) | Trả về giá trị số a làm tròn lên |
| floor(a) | Trả về giá trị số a làm tròn xuống |
| max(a,b) | Trả về giá trị lớn hơn trong 2 số a,b |
| min(a,b) | Trả về giá trị nhỏ hơn trong 2 số a,b |

1. **ArrayList**

* Khởi tạo một arraylist

ArrayList<Double> list = new ArrayList<>();

ArrayList<Integer> list = new ArrayList<>();

ArrayList<String> list = new ArrayList<>();

* Các phương thức

| Phương thức | Chức năng |
| --- | --- |
| list.add(a); | Thêm phần tử a vào list |
| list.clear(); | Xóa hết phần tử trong list |
| list.contains(a); | Kiểm tra xem trong list có chứa a hay không |
| list.indexOf(a) | Kiểm tra xem a ở vị trí nào trong list |
| list.isEmpty() | Kiểm tra xem list có rỗng hay không |
| list.remove(a) | Xóa phần tử a, hoặc phần tử ở vị trí a |
| list.size() | Kiểm tra xem trong list có bao nhiêu phần tử |
| list.sublist(a,b) | Lấy phần tử từ vị trí a đến b trong list |
| list.removeAll(list) | Xóa hết phần tử trong list |
| list.get(i) | Lấy phần tử thứ i trong list |
| list.get(i).equalsIgnoreCase(a) | So sánh phần tử thứ i với a(không phân biệt chữ hoa chữ thường) |
| list.get(i).equals(a) | So sánh phần tử thứ i với a(có phân biệt chữ hoa chữ thường) |
| list.get(i).contains(“a”) | Kiểm tra xem trong phần tử nào trong list có chứa a |
| list.get(i).endsWith("a") | Kiểm tra xem phần tử nào trong list kết thúc bằng chữ a |
| list.get(i).length() | Kiểm tra xem phần tử i trong list có độ dài bao nhiêu |
| list.get(i).startsWith("a") | Kiểm tra xem phần tử nào trong list bắt đầu bằng chữ a |
| list.get(1).substring(0,2) | Lấy kí kí tự 0-2 của phần tử thứ 1 trong list |

1. **Collections.**

| Tên phương thức | Chức năng |
| --- | --- |
| Collections.sort(list) | Sắp xếp list tăng dần theo tên |
| Collections.reverse(list); | Đảo list ngược lại |
| Collections.swap(list, a, b); | Đổi vị trí 2 index a và b trong list |

1. **Các hàm có sẵn khác**

* Character
* Character.isDigit(i): Kiểm tra xem i có phải 1 số không
* Character.isAlphabetic(i): Kiểm tra xem i có phải chữ thuộc bảng chữ cái không
* Character.isLetter(i): Kiểm tra xem i có phải 1 chữ cái không
* Character.isLowerCase(i): Kiểm tra xem i có viết thường không
* Character.isUpperCase(i): Kiểm tra xem i có viết hoa không
* Integer
* Double

1. **Code mẫu để copy :))) (Tăng,giảm,max,min,change index,...)**

* Tìm max

int max=t.get(0).getSize(); //gán max = giá trị đầu tiên của list

int pos=0; // gọi 1 biến pos để lấy vị trí của max.

for (int i=0;i<t.size();i++) // duyệt toàn bộ list

{

if (t.get(i).getSize()>max) // nếu có 1 phần tử nào lớn hơn max thì …

{

max=t.get(i).getSize(); //gán max bằng phần tử đó

pos=i; //gán vị trí mới cho max

}

}

* Tìm min

int min=list.get(0).getLeg();

int pos2=0;

for (int i=0;i<list.size();i++){

if (list.get(i).getLeg() < min){

min = list.get(i).getLeg();

pos2 = i;

}

}

* Sắp xếp tăng/giảm dần

Cách 1: Cách này thuần C hơn :)) Code này sx tăng dần

* Bien kieu int,double,char...

for (int i=0;i<list.size();i++){

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

if (list.get(i).getLeg() > list.get(j).getLeg()){ // giảm dần thì sửa dấu > thành dấu < là được nhé =))

Collections.swap(list, i, j);

}

}

}

* Bien kieu String

for (int i=0;i<t.size();i++)

{

for (int j=i+1;j<t.size();j++)

{

if (t.get(i).getName().compareTo( t.get(j).getName())>0)

// giảm dần thì sửa dấu > thành dấu < là được nhé =))

{

Collections.swap(t, i, j);

}

}

}

Cách 2: Cách này thuần java hơn :))) 2 code dưới là sắp xếp tăng dần. Giảm dần thì thêm dấu trừ trước lệnh return là được..

+ Dùng compareTo cho biến kiểu String

Collections.sort(list, new Comparator<Cat>(){

@Override

public int compare(Cat o1, Cat o2) {

if ((o1.getArea()).compareTo(o2.getArea()) != 0){

return (o1.getArea()).compareTo(o2.getArea());

}

return 0;

}

});

* Dùng dấu - cho biến kiểu int,double,char,..

Collections.sort(list, new Comparator<Cat>(){

@Override

public int compare(Cat o1, Cat o2) {

if ((o1.getLeg())-(o2.getLeg()) != 0){

return o1.getLeg()-(o2.getLeg());

}

return 0;

}

});

- Kiểm tra xem có phần tử nào trong list chứa …. không.

Chứa thì dùng lệnh contains nhen :))

int count=0;

for (int i=0;i<list.size();i++){

if (list.get(i).getArea().contains("A")

|| list.get(i).getArea().contains("B"))

count+=1;

}

return count;

Nếu bài hỏi không chứa … thì thêm dấu ! vào

int count=0;

for (int i=0;i<t.size();i++)

{

if (!(t.get(i).getName().contains("B") &&t.get(i).getName().contains("A") ))

count++;

}

return count;

* Tính trung bình cộng

int count=0;

int dem=0;

for (int i=0;i<t.size();i++)

{

count+=t.get(i).getColor();

}

return count/t.size();

NOTE: KHI MUỐN PHỦ ĐỊNH THÌ THÊM DẤU “!”

Q4

TRƯỜNG HỢP 1 : KHI BÀI YÊU CẦU COUNT VÀ TÁCH CÁC TỪ BỞI KHOẢNG TRẮNG

public boolean check(String n){

int count=0;

for (int i=0;i<n.length();i++)

{

if (Character.isDigit(n.charAt(i)) && n.charAt(i)>='0' && n.charAt(i)<='9' )

count++;

}

return count>0;

}

@Override

public int f1(String str) {

int count=0;

String words[] = str.split(" ");

for (int i=0;i<words.length;i++)

{

if (check(words[i]))

{

count++;

}

}

return count;

}

@Override

public int f1(String str) {

int count=0;

String words[] = str.split(" ");

for (int i=0;i<words.length;i++)

{

if (words[i].endsWith("m"))

count++;

}

return count;

}

TRƯỜNG HỢP 2: KHI ĐỀ BÀI YÊU CẦU COUNT NHƯNG KHÔNG PHẢI TÁCH CHUỖI TẠI KHOẢNG TRẮNG

public int f1(String str) {

int count=0;

for (int i=0;i<str.length();i++)

{

if (Character.isDigit(str.charAt(i)) && str.charAt(i)>='0' && str.charAt(i)<='9' && str.charAt(i)%2!=0)

count+=1;

// count+=str.charAt(i)-’0’;

}

return count;

}