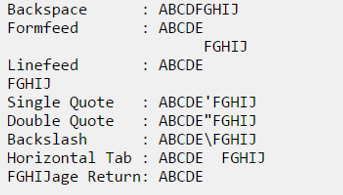
EXERCISES

# Session 3

## Bài tập 1

Trong class **Sequence**, viết chương trình trong phương thức main in ra các định dạng sau



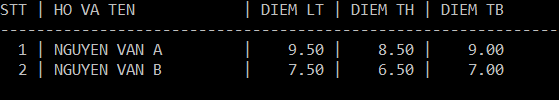
## Bài tập 2

Trong class **AddNumber**, viết chương trình trong phương thức main thực hiện các bước sau

* Khai báo các biến cục bộ integer num1, num2
* Gán các giá trị lần lượt 5, 6
* In ra tổng, hiệu, tích, thương của 2 số đó

## Bài tập 2.1

Xây dựng lớp PrintFormat, với phương thức main để in ra định dạng sau, sử dụng các kiểu định dạng đã học



## Bài tập 3

Trong class **CalculateNumber**, viết chương trình thực hiện các bước sau trong phương thức main

* Viết chương trình cho nhập 2 số nguyên từ bàn phím
* Tính giá trị tổng, hiệu, tích, thương của 2 số nguyên đó

## Bài tập 2.2

Viết chương trình cho nhập vào bán kính đường tròn, từ đó tính ra chu vi và diện tích hình tròn. Kết quả lấy 2 chữ số thập phân

## Bài tập 4.1

Xây dựng lớp **PrintCharacter**, ở phương thức main, ta in ra ký tự của mã ASCII được nhập vào (mã này số nguyên). Sử dụng System.out.printf(“%c”, c); c kiểu int

## Bài tập 4.2

Viết chương trình nhập vào ký tự và xuất ra mã ASCII của nó.

Ở đây ta có ép kiểu từ kiểu char sang kiểu int.

Muốn nhập được ký tự, ta sử dụng lệnh sau : c = scanner.next().charAt(0) ;

## Bài tập 4

Trong class **MaxMin**, viết chương trình thực hiện các bước sau trong phương thức main

* Thông báo nhập vào 3 số nguyên: num1, num2, num3
* In ra số lớn hơn và số nhỏ hơn của 2 số num1 và num2 sử dụng toán tử 3 ngôi
* In ra trị tuyệt đối của num3 sử dụng toán tử 3 ngôi

## Bài tập 5

~~class UnaryDemo {~~

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

~~int result = +1;~~

~~// result is now 1~~

~~System.out.println(result);~~

~~result--;~~

~~// result is now 0~~

~~System.out.println(result);~~

~~result++;~~

~~// result is now 1~~

~~System.out.println(result);~~

~~result = -result;~~

~~// result is now -1~~

~~System.out.println(result);~~

~~boolean success = false;~~

~~// false~~

~~System.out.println(success);~~

~~// true~~

~~System.out.println(!success);~~

~~}~~

}

## Bài tập 6

~~class PrePostDemo {~~

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

~~int i = 3;~~

~~i++;~~

~~// prints 4~~

~~System.out.println(i);~~

~~++i;~~

~~// prints 5~~

~~System.out.println(i);~~

~~// prints 6~~

~~System.out.println(++i);~~

~~// prints 6~~

~~System.out.println(i++);~~

~~// prints 7~~

~~System.out.println(i);~~

~~}~~

~~}~~

# Session 4

## Bài tập 1.4

Viết chương trình cho người dùng nhập vào 1 ký tự. Nếu người dùng nhập vào ký tự ‘A’ hoặc ‘a’ thì in ra bạn đã nhập vào ký tự ‘A’ hoặc ‘a’.

## Bài tập 1.3

Viết chương trình cho người dùng nhập vào 1 số nguyên. Nếu số nguyên đó chẵn thì in ra “So nhap vao la so chan”, sử dụng toán tử % để xét chia hết cho 2.

## Bài tập 1.2

Viết chương trình tìm số lớn nhất trong 3 số thực a, b và c được nhập vào. Hướng dẫn: ban đầu gán max = a; sau đó xét các điều kiện b>=max, c>=max hay không, sau đó gán max tương ứng?

## Bài tập 1 (if else)

Trong class **CheckScore**, viết chương trình thực hiện trong phương thức main các bước sau:

* Thông báo nhập vào điểm số 1 sinh viên
* Nếu điểm số >= 10 in ra PASS, không thì in FAIL

## Bài tập 1.1 (if else)

Write Java program to allow the user to input his/her age. Then the program will show if the person is eligible to vote. A person who is eligible to vote must be older than or equal to 18 years old.

## Bài tập 2 (if else)

Trong class **CheckEvenOdd**, viết chương trình thực hiện trong phương thức main các bước sau:

* Thông báo nhập vào 1 số nguyên
* Xuất ra kết quả số đó là số chẵn hay lẻ

## Bài tập 2.1 (nested if)

Giải phương trình bậc 1.

## Bài tập 3 (if-else-if)

Trong class **ScoreLevel**, viết chương trình thực hiện trong phương thức main các bước sau:

* Thông báo nhập vào điểm số sinh viên vào biến score
* Nếu điểm số <10, in ra “FAIL”, 10<=score<15, in ra “C”, 15<=score<18 in ra “B”, 18<=score<=20: “A”

*(Gợi ý: - sử dụng cấu trúc if-else-if)*

## Bài tập 3.1

Viết chương trình nhập vào 1 số, kiểm tra số nhập vào có là số âm, số dương hay bằng 0, sử dụng if-else-if để xét dần từng trường hợp.

## Bài tập 3.2

Thông báo nhập vào 1 điểm số.

Sử dụng if-else-if để xét:

* Nếu điểm số >=9, in ra “XUAT SAC”
* Nếu điểm số >=8, (<9), in ra “GIOI”
* Nếu điểm số >=6.5, in ra “KHA”
* Nếu điểm số >=5, in ra “TRUNG BINH”

Còn lại, in ra “YEU”

## Bài tập 3.3

Viết chương trình tính lương của nhân viên dựa vào thâm niên công tác (TNCT) như sau: **Lương = hệ số\*lương cơ bản,** trong đó **lương cơ bản = 650000 đ.**

Nếu TNCT<12 tháng: hệ số = 1.92;

Nếu 12 <= TNCT < 36 tháng: hệ số = 2.34;

Nếu 36 <= TNCT < 60 tháng: hệ số = 3;

Nếu TNCT >= 60 tháng: hệ số = 4.5

## Bài tập 3.4

Thông báo cho người dùng nhập vào 3 cột điểm: điểm chuyên cần (dcc), điểm giữa kỳ (dgk), điểm cuối kỳ (dck). Điểm trung bình (dtb) được tính như sau:

*Dtb = (dcc\*0.1 + dgk\*0.3 + dck\*0.6).*

Nếu điểm trung bình dtb>=9 thì in ra loại A;

Nếu 7 <= dtb < 9 thì in ra loại B;

Nếu 5 <= dtb < 7 thì in ra loại C;

Nếu dtb <5 thì in ra loại D.

## Bài tập 3.5

Viết chương trình giải phương trình bậc 2, sử dụng cách biện luận đã học để xét các trường hợp tham số a, b, c (sau khi đã đi qua phần if lồng nhau và if-else-if)

## Bài tập 3.6

Sử dụng cấu trúc switch-case để kiểm tra tính chẵn lẻ của 1 số nguyên nhập vào.

## Bài tập 4 (if else if, switch-case)

Trong class **PrintNumberInWord**, viết chương trình thực hiện trong phương thức main các bước sau:

* Thông báo nhập vào 1 số nguyên
* Nếu số đó = 1, thì in ra “ONE”
* Nếu số đó = 2, thì in ra “TWO”
* …
* Nếu số đó = 9, thì in ra “NINE”

*(Gợi ý: - sử dụng cấu trúc switch-case)*

## Bài tập 5 (if else, switch-case)

Xây dựng class PrintDays, cho nhập vào tháng và năm, in ra số ngày của tháng của năm đó.

*(Gợi ý: - sử dụng cấu trúc switch-case*

* *Năm nhuận tháng 2 có 29 ngày, năm nhuận là năm chia hết cho 4 và ko chia hết cho 100 hoặc là chia hết cho 400. Căn cứ vào đó để xét if else trong switch case)*

## Bài tập 5.1

Viết chương trình để thể hiện khả năng của máy tính. Người dùng nhập vào một ký tự và chương trình sẽ hiển thị ngôn ngữ tương ứng. Sử dụng switch-case để chọn và hiển thị thông báo. Sử dụng default để hiển thị thông báo “**Bạn đã nhập sai**” trong trường hợp người dùng nhập không khớp với các ký tự như đã mô tả ở cột **đầu vào** của bảng sau

|  |  |
| --- | --- |
| **Đầu vào (Input)** | **Đầu ra (Output)** |
| A hoặc a | Ada |
| B hoặc b | Basic |
| C hoặc c | Cobol |
| D hoặc d | Android |
| F hoặc f | Fortran |
| W hoặc w | Windows Phone |

## Bài tập 5.2

Viết chương trình tính lương thực lãnh của nhân viên, biết rằng **lương thực lãnh = lương + phụ cấp**. Trong đó **phụ cấp** sẽ dựa vào **thứ hạng của nhân viên.**

Nhập lương, thứ hạng và tính lương thực lãnh của nhân viên. Cho bảng phụ cấp dựa vào thứ hạng

|  |  |
| --- | --- |
| **Thứ hạng của nhân viên** | **Phụ cấp** |
| A | 300 |
| B | 200 |
| C | 100 |

## Bài tập 5.3

Yêu cầu người thực hiện chương trình nhập vào tháng trong năm. Chương trình sẽ in ra số ngày tương ứng của tháng đó. Nếu là tháng 2 được nhập vào thì in ra, tháng 2 có 28 hoặc 29 ngày. Sử dụng if-else-if hay switch-case

## Bài tập 5.4

Thông báo nhập vào 1 số.

Sử dụng switch case để xét:

* Nếu số đó = 1, 2 hoặc 3, in ra “XUAN”
* Nếu số đó = 4, 5 hoặc 6, in ra “HA”
* Nếu số đó = 7, 8 hoặc 9, in ra “THU”
* Nếu số đó = 10, 11 hoặc 12, in ra “DONG”
* Ngoài ra, sẽ in ra “KHONG HOP LE”

# Session 5

## Bài tập 1.1

In ra 20 lần chữ “Hello Java Programming” sử dụng vòng lặp while.

## Bài tập 1.2

In ra các số chẵn từ 1 đến 100 sử dụng vòng lặp while.

## Bài tập 1.3

In ra bảng cửu chương 5 sử dụng vòng lặp while.

## Bài tập 1.4

Viết chương trình in ra màn hình dãy các số nguyên chẵn nhỏ hơn 100 và lớn hơn 2 sử dụng vòng lặp While. Có dạng (Day so nguyen: 4 6 8 10 …)

## Bài tập 1 (while)

Viết chương trình để in ra từ 1 đến 100. In ra 10 số trên 1 dòng

## Bài tập 2 (while)

Viết chương trình để in ra bảng ký tự tương ứng với mã ASCII từ 1 đến 122.

## Bài tập 3 (while)

Viết chương trình để thực hiện các công việc sau:

* Cung cấp phép toán cộng giữa 2 số nguyên ngẫu nhiên từ 0 đến 9.
* Nếu người dùng đưa ra đáp án sai, chương trình cho nhập lại
* Đến khi nào người dùng đưa ra đáp án đúng thì chương trình dừng lại

*Gợi ý: - sử dụng cú pháp int number1=(int)(Math.random()\*10) để tạo ra số ngẫu nhiên từ 0 đến 9*

## Bài tập 6 (do while)

Viết chương trình để thực hiện các công việc sau:

* In ra màn hình yêu cầu nhập 1 số nguyên
* Chương trình sẽ lặp mãi cho đến khi nào người dùng nhập giá trị 0.
* Chương trình sẽ tính tổng các số nguyên nhập vào đó

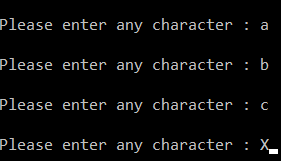
## Bài tập 7 (do while)

Viết chương trình để thực hiện các công việc sau:

* In ra màn hình yêu cầu nhập 1 số nguyên
* Chương trình sẽ in ra màn hình số nhập vào là số chẵn hay lẻ
* Chương trình sẽ lặp mãi cho đến khi nào người dùng nhập giá trị 0.

## Bài tập 7.1

Viết chương trình thông báo cho người dùng nhập vào 1 ký tự, sau đó in ra ký tự người dùng vừa nhập vào. Chừng nào người dùng còn nhập ký tự khác ‘X’ thì cho nhập tiếp, ngược lại dừng nhập và kết thúc chương trình. Sử dụng do..while.



## Bài tập 8 (for)

Viết chương trình để thực hiện các công việc sau:

* In ra màn hình 100 dòng “Welcome to JAVA”

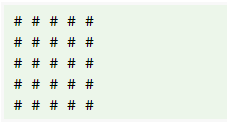
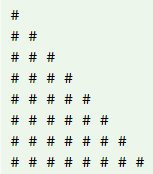
## Bài tập 9 (for)

Viết chương trình để thực hiện các công việc sau:

* Nhập vào 5 số nguyên
* Tính tổng của 5 số nguyên đó

## Bài tập 10 (nested for)

Viết chương trình để in ra màn hình dạng sau:

## Bài tập 4 (while, break)

Viết chương trình để kiểm tra 1 chuỗi nhập vào có đối xứng hay không? (palindrome)

*Gợi ý: - Dùng phương thức nextLine() của đối tượng Scanner để nhập chuỗi*

*- s.length() để trả về số ký tự trong chuỗi*

*- sử dụng s.charAt(index) để truy cập tới ký tự tại chỉ số nào đó của chuỗi*

*- cho chỉ số low, high để quét đầu và cuối chuỗi nhập vào, low++, high-- sau mỗi lần kiểm tra*

*if (s.charAt(low)!=s.charAt(high)){*

*isPalindrome =false;*

*break;*

*}*

## Bài tập 12 (for)

Viết chương trình để thực hiện các bước công việc sau:

* Thông báo người dùng nhập vào 1 số nguyên **num1**
* In ra số đó có phải là số nguyên tố hay ko

## Bài tập 13 (for)

Viết chương trình để thực hiện các bước công việc sau:

* Thông báo người dùng nhập vào 1 số nguyên **num1**
* In ra số đó có phải là số chính phương hay ko

## Bài tập 5 (for, continue)

Viết chương trình để tính tổng các số từ 1 đến 150, trừ ra các số 33, 56, 69 (không tính vào tổng)

## Bài tập 11 (while, for, break)

Viết chương trình để thực hiện các bước công việc sau:

* Thông báo người dùng nhập vào 1 số nguyên ***num1***
* In ra các số nguyên tố trong khoảng từ 1 đến số nguyên nhập vào ***num1***

*Gợi ý:*

* *số nguyên tố là số chỉ chia hết cho 1 và chính nó*
* *Sử dụng toán tử % (chia lấy dư) để xác định tính chia hết*

# Session 6

## Bài tập 1

Viết lớp **Rectangle** gồm có 2 thuộc tính là **width**và **height** kiểu int, access modifier kiểu mặc định. Trong lớp này, viết phương thức main thực hiện các công việc sau:

* Khởi tạo đối tượng **Rectangle**, sử dụng toán tử **new**
* Thiết lập các thuộc tính **width** và **height** cho đối tượng đó dùng toán tử ..
* Sau đó, in ra các thuộc tính **width** và **height** đã được thiết lập

## Bài tập 2

Viết lớp **Rectangle** gồm có 2 thuộc tính kiểu int, access modifier kiểu mặc định là **width**và **height**. Trong lớp này, viết các phương thức public để thiết lập và hiển thị các thuộc tính này, gồm có **setWidth(int width1)**, **setHeight(int height1)**, **displayInfo()**. Cũng trong lớp này, viết phương thức ***main*** thực hiện các công việc sau:

* Khởi tạo 2 đối tượng **Rectangle**, sử dụng toán tử **new**
* Thiết lập các thuộc tính **width** và **height** cho từng đối tượng đó sử dụng **setWidth()** và **setHeight()**.
* Sau đó, in ra các thuộc tính **width** và **height** đã được thiết lập của từng đối tượng thông qua phương thức **displayInfo()**.

## Bài tập 2.1

Viết lớp **Circle** gồm có 1 thuộc tính kiểu int tên là **radius**, access modifier kiểu mặc định. Trong lớp này, viết các phương thức public để thiết lập và hiển thị các thuộc tính này, gồm có **setRadius(int radius1)**, **displayInfo()**. Cũng trong lớp này, viết phương thức ***main*** thực hiện các công việc sau:

* Khởi tạo 2 đối tượng **Circle**, sử dụng toán tử **new**
* Thiết lập thuộc tính **radius** cho từng đối tượng đó sử dụng **setRadius().**
* Sau đó, in ra thông tin radius đã được thiết lập của từng đối tượng thông qua phương thức **displayInfo()**.

## Bài tập 3 (no-args constructor)

Viết lớp **Circle** gồm có thuộc tính **radius** kiểu double, access modifier kiểu mặc định. Trong lớp này, viết phương thức khởi tạo (không tham số) để khởi tạo giá trị cho thuộc tính **radius** (ví dụ cho **radius** = 20). Viết phương thức **main**trong lớp **TestCircle** để sử dụng đối tượng **Circle**này, và in ra giá trị của thuộc tính **radius** của đối tượng đó.

## Bài tập 3.1 (no-args constructor)

Viết lớp **Circle** gồm có thuộc tính **radius** kiểu double, access modifier kiểu mặc định. Trong lớp này, viết phương thức khởi tạo (không tham số) để khởi tạo giá trị cho thuộc tính **radius** (ví dụ cho **radius** = 20). Xây dựng phương thức displayInfo() để hiển thị thuộc tính radius của đối tượng. Viết phương thức **main**trong lớp **TestCircle** để sử dụng đối tượng **Circle**này, và in ra giá trị của thuộc tính **radius** của đối tượng đó thông qua phương thức displayInfo().

## Bài tập 4 (parameterized constructor)

Viết lớp **Circle** gồm có thuộc tính là **radius,** kiểu double, access modifier kiểu mặc định. Trong lớp này, viết phương thức khởi tạo không tham số (radius = 20) và có tham số để khởi tạo giá trị cho thuộc tính **radius**. Viết phương thức **main**trong lớp **TestCircle** để sử dụng đối tượng **Circle**này, sử dụng 2 constructor đã xây dựng nên 2 đối tượng Circle khác nhau và in ra giá trị của thuộc tính **radius** của 2 đối tượng đó.

## Bài tập 5 (reference)

Viết lớp **Circle** gồm có thuộc tính là **radius**, kiểu double, access modifier kiểu mặc định. Trong lớp này, viết phương thức khởi tạo ko tham số (radius = 20) và khởi tạo có tham số để khởi tạo giá trị cho thuộc tính **radius***,* viết phương thức **displayInfo()** để hiển thị thông tin thuộc tính đối tượng. Viết phương thức **main**trong lớp **TestCircle** gồm các bước thực hiện sau :

* Khai báo và khởi tạo không tham số đối tượng **Circle** và được tham chiếu bởi biến **circle1**
* Khai báo và khởi tạo có tham số đối tượng **Circle** và được tham chiếu bởi biến **circle2**
* Khai báo biến **circle3** và tham chiếu đến cùng đối tượng mà **circle2** tham chiếu tới
* In ra các thông tin của đối tượng mà **circle1**, **circle2**, **circle3** tham chiếu tới, sử dụng phương thức **displayInfo()** đã xây dựng.

## Bài tập 6 (private)

Viết lớp **Circle** gồm có thuộc tính **private** **radius**, kiểu double. Trong lớp này, viết phương thức khởi tạo ko có tham số (**radius** = 20) và khởi tạo có tham số để khởi tạo giá trị cho thuộc tính **radius***,* viết phương thức thiết đặt thuộc tính **setRadius()**, tính chu vi **displayPerimeter()**, phương thức tính diện tích **displayArea()**.

Viết lớp **TestCircle**, trong lớp này, xây dựng phương thức **main** gồm các bước công việc sau :

* Khởi tạo đối tượng **circle1** không có tham số.
* Khởi tạo đối tượng **circle2** có tham số.
* Khởi tạo đối tượng **circle3** không tham số, sau đó, set lại giá trị thuộc tính **radius** = 60.
* In ra chu vi và diện tích của 3 đối tượng **circle1**, **circle2**, **circle3** bằng cách gọi các phương thức **displayPerimeter()** và **displayArea()**.

# Session 7

## Bài tập 1.3

~~Viết chương trình nhập vào chữ hoa và chuyển đổi nó sang ký tự chữ thường. Xây dựng phương thức convertUpperCase(char c) trong lớp~~ **~~CharThings~~**~~, sau đó gọi nó từ hàm main~~

## Bài tập 1.3.1

~~Viết chương trình nhập vào chữ hoa và chuyển đổi nó sang ký tự chữ thường. Xây dựng phương thức convertUpperCase(char c) trong lớp~~ **~~CharThings~~**~~, phương thức này trả về ký tự hoa tương ứng với ký tự thường c. Phương thức này sẽ được gọi từ hàm main.~~

## Bài tập 1.3 và 1.3.1

Viết phương thức xác định số nguyên tố, làm 2 trường hợp có trả về và ko có trở về

## Bài tập 1.1

Tạo 1 lớp Calculator, xây dựng 2 phương thức

* Phương thức max(int a, int b, int c) trả về giá trị lớn nhất trong 3 số a, b, c trên.
* Phương thức tính trung bình cộng của 3 số a, b, c: average(int a, int b, int c).
* Trong phương thức main, gọi tới 2 phương thức này để sử dụng, sau khi nhập vào 3 số a, b, c

Write a Java method to find the smallest number among three numbers. [Go to the editor](http://www.w3resource.com/java-exercises/method/index.php#editorr)  
Test Data:  
Input the first number: 25  
Input the Second number: 37  
Input the third number: 29   
Expected Output:

The smallest value is 25.0

Write a Java method to compute the average of three numbers. [Go to the editor](http://www.w3resource.com/java-exercises/method/index.php#editorr)  
Test Data:  
Input the first number: 25   
Input the second number: 45  
Input the third number: 65  
Expected Output:

The average value is 45.0

## Bài tập 1.2

Tạo 1 lớp StringManipulation, với các phương thức:

* Phương thức checkCharacter(String str, char a) trả về true nếu a tồn tại trong chuỗi str, và false nếu ngược lại.
  + *Gợi ý:* 
    - *Sử dụng vòng lặp for để duyệt chuỗi*
    - *Sử dụng str.lenghth() để trả về số ký tự trong chuỗi*
    - *Sử dụng charAt(i) để trả về ký tự nào đó trong chuỗi*
* Phương thức countVowel(String str) trả về số các nguyên âm tồn tại trong chuỗi str.
* Viết phương thức main để gọi phương thức checkCharacter để kiểm tra ký tự có tồn tại trong chuỗi nào đó và countVowel để đếm số nguyên âm có trong chuỗi.

Write a Java method to count all vowels in a string

1. **import** java.util.Scanner;
2. **public** **class** Exercise4 {
3. **public** **static** **void** main(String[] args)
4. {
5. Scanner in = **new** Scanner(System.in);
6. System.out.print("Input the string: ");
7. String str = in.nextLine();
9. System.out.print("Number of  Vowels in the string: " + count\_Vowels(str)+"\n");
10. }
11. **public** **static** **int** count\_Vowels(String str)
12. {
13. **int** count = 0;
14. **for** (**int** i = 0; i < str.length(); i++)
15. {
16. **if** (str.charAt(i) == 'a' || str.charAt(i) == 'e' || str.charAt(i) == 'i'
17. || str.charAt(i) == 'o' || str.charAt(i) == 'u')
18. {
19. count++;
20. }
21. }
22. **return** count;
23. }
24. }

## Bài tập 1.2.1

Tạo 1 lớp Prime, xây dựng phương thức boolean isPrime(int number) để trả về true, nếu number là số nguyên tố, và trả về false, nếu number không phải là số nguyên tố. Từ main gọi đến phương thức này để kiểm tra.

## Bài tập 1.2.2

Tạo 1 lớp Square, xây dựng phương thức boolean isSquare(int number) để trả về true, nếu number là số chính phương, và trả về false, nếu number không phải là số chính phương. Từ main gọi đến phương thức này để kiểm tra.

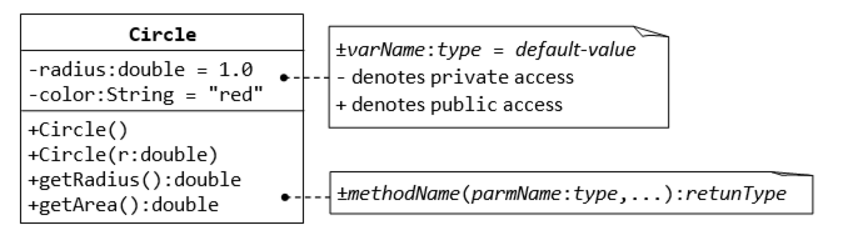
## Bài tập 1 (default, parameterized constructor, return)

Viết lớp **Circle** gồm có thuộc tính **private** **radius**, kiểu double. Trong lớp này, viết phương thức khởi tạo mặc định không tham số (**radius** = 20) và khởi tạo có tham số để khởi tạo giá trị cho thuộc tính **radius***,* viết phương thức thiết đặt thuộc tính **setRadius()**, phương thức tính chu vi **getPerimeter()** trả về giá trị chu vi của đường tròn, phương thức tính diện tích **getArea()** trả về giá trị diện tích hình tròn.

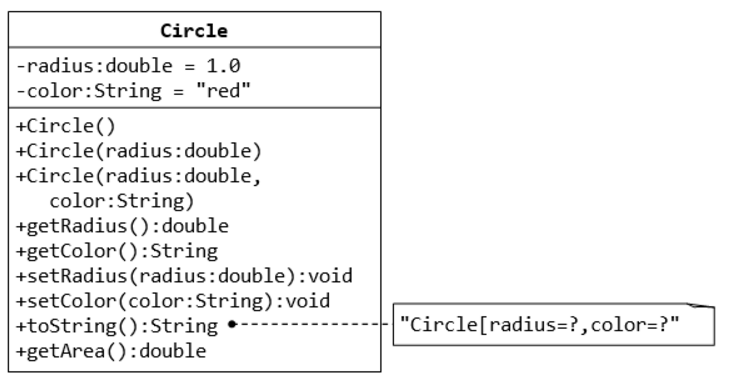
Viết lớp **TestCircle**, trong lớp này, xây dựng phương thức **main** gồm các bước công việc sau :

* Khởi tạo đối tượng **circle1** không có tham số.
* Khởi tạo đối tượng **circle2** có tham số.
* Khởi tạo đối tượng **circle3** không tham số, sau đó, set lại giá trị thuộc tính **radius** = 40.
* In ra chu vi và diện tích của 3 đối tượng **circle1**, **circle2**, **circle3** bằng cách gọi các phương thức **getPerimeter()** và **getArea()**.

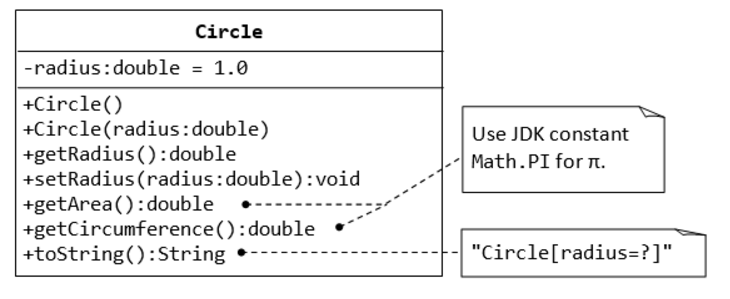
## Bài tập 7



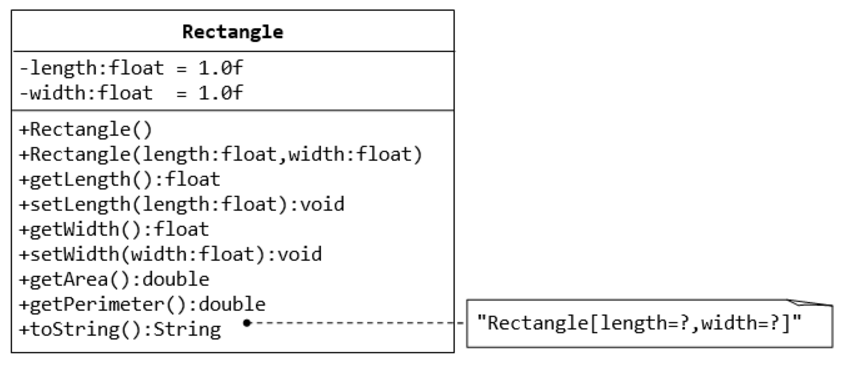
## Bài tập 8



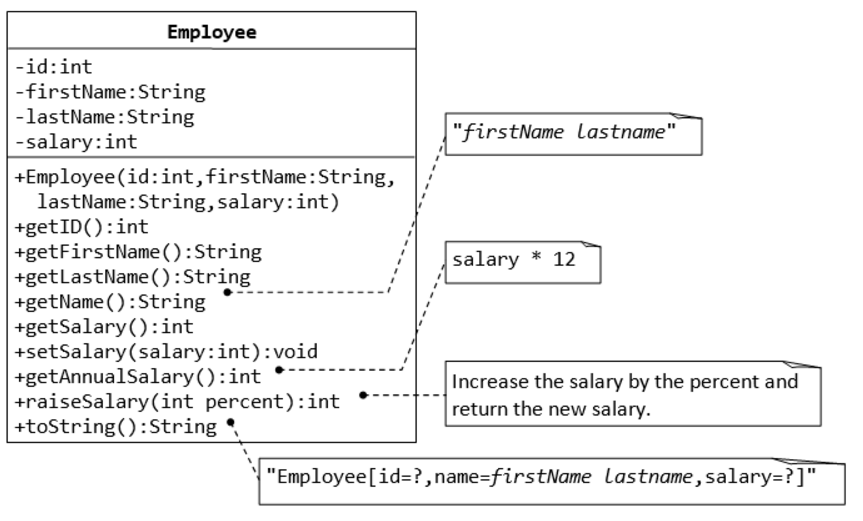
## Bài tập 9



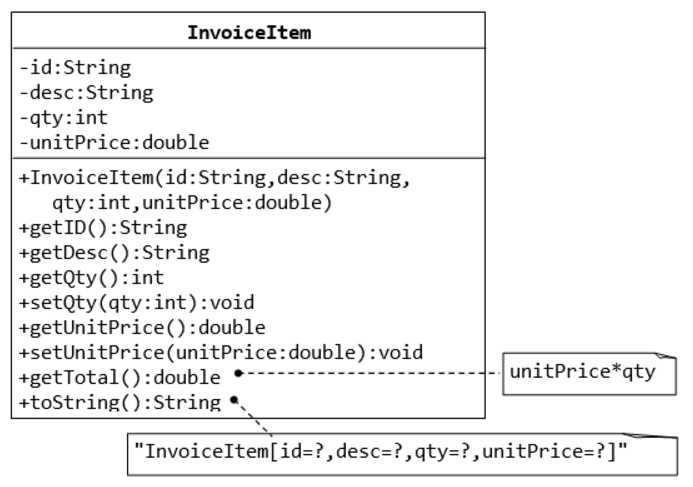
## Bài tập 10



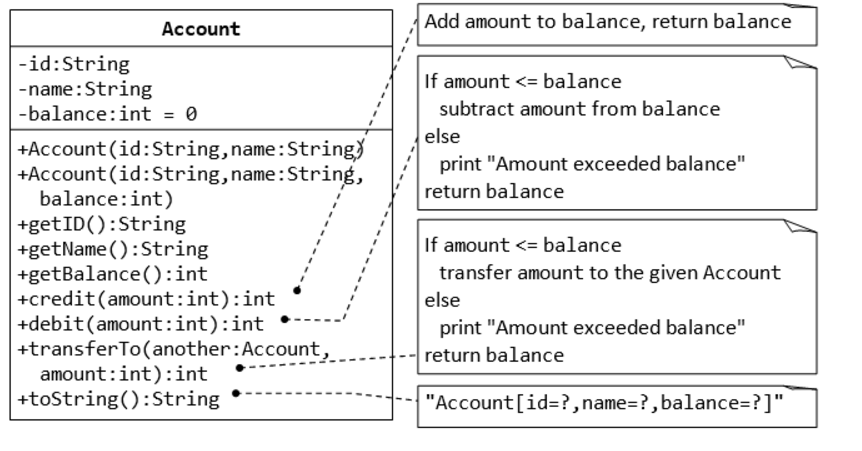
## Bài tập 11



## Bài tập 12



## Bài tập 13



## Bài tập 2 (overloading)

Viết lớp **MaxMin** gồm có 4 phương thức **max(int, int), min(int, int), max(double, double), min(double, double)** trả về các giá trị max, min tương ứng với 2 số nhập vào. Viết lớp TestMaxMin và trong lớp này, viết phương thức **main** thực hiện tính max, min của 2 số nhập vào, sử dụng các phương thức đã xây dựng ở trên.

# Session 8

**Mảng 1 chiều**

## Bài tập 1 (array, for)

Xây dựng lớp ArrayThings, gồm có các đặc điểm sau:

* Có 1 trường là mảng số nguyên
* 1 phương thức để nhập các giá trị cho các phần tử của mảng
* 1 phương thức in ra mảng
* 1 phương thức tính tổng các phần tử của mảng

~~Viết chương trình tính tổng các số nhập vào, cụ thể như sau:~~

* ~~Thông báo nhập vào số phần tử cần nhập vào.~~
* ~~Tiến hành nhập vào 1 mảng, sử dụng đối tượng Scanner~~
* ~~Sau đó in ra các phần tử đã nhập vào mảng, và tính tổng các phần tử đó~~

## Bài tập 2 (array, for)

Xây dựng lớp ArrayThings, gồm có các đặc điểm sau:

* Có 1 trường là mảng số nguyên
* 1 phương thức để nhập các giá trị cho các phần tử của mảng
* 1 phương thức in ra mảng
* 1 phương thức tính trung bình tổng các phần tử của mảng
* 1 phương thức tính số phần tử có giá trị lớn hơn hay bằng giá trị trung bình đó

~~Viết chương trình nhập vào các số và lưu vào mảng, sau đó in ra các số đó, tính trung bình tổng các số đó, và tính số phần tử có giá trị lớn hơn hay bằng giá trị trung bình đó.~~

**Mảng nhiều chiều**

## Bài tập 3 (array, for)

Xây dựng lớp Matran, có các đặc điểm sau:

* 1 thuộc tính private là mảng 2 chiều kiểu int
* 1 phương thức nhập các phần tử vào mảng 2 chiều ở trên
* 1 phương thức in ra mảng 2 chiều ở trên
* ~~1 phương thức tính tổng các số trong mỗi cột~~.

## Bài tập 4 (array, for)

Cộng ma trận

**ArrayList**

## Bài tập 5.2

**Làm lại bài tập 1, 2 của Session này sử dụng ArrayList thay vì array thông thường.**

## Bài tập 5 (ArrayList, while)

Xây dựng lớp ArrayListThings, có các đặc điểm sau:

* 1 thuộc tính private kiểu ArrayList để chứa mảng các số nguyên
* Xây dựng phương thức inputArrayList() để nhập các giá trị vào mảng ArrayList đến khi nhập 0 thì dừng lại
* Xây dựng phương thức printArrayList() để in các giá trị trong mảng ArrayList ra
* Xây dựng phương thức tính trung bình tổng các phần tử lưu trong mảng ArrayList
* ~~Xây dựng phương thức tính số phần tử có giá trị lớn hơn hay bằng giá trị trung bình đó.~~

## Bài tập 5.1 (ArrayList)

Write a Java program to iterate through all elements in a array list

import java.util.\*;

public class Exercise2 {

public static void main(String[] args) {

// Creae a list and add some colors to the list

List<String> list\_Strings = new ArrayList<String>();

list\_Strings.add("Red");

list\_Strings.add("Green");

list\_Strings.add("Orange");

list\_Strings.add("White");

list\_Strings.add("Black");

// Print the list

for (String element : list\_Strings) {

System.out.println(element);

}

}

}

## Bài tập 6 (ArrayList, while)

* Xây dựng lớp HocSinh gồm có các trường ten (String), tuoi (int) cùng các getters/setters.
* Xây dựng lớp HocSinhManager, gồm có các đặc điểm sau:
  + 1 trường private là mảng đối tượng ArrayList<HocSinh>
  + Phương thức nhapHocSinh() để nhập các phần tử HocSinh vào mảng, đến khi trường tuổi = 0 thì ngừng nhập, sử dụng vòng lặp while
  + Phương thức xuatHocSinh() để xuất các phần tử HocSinh từ mảng ra.
* Sau khi đã nhập vào ArrayList, bây giờ ta xuất ra danh sách các đối tượng HocSinh được nhập vào

package javaandroidvn;  
 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.io.Serializable;import java.util.ArrayList;import java.util.Scanner;import java.util.logging.Level;import java.util.logging.Logger;  
  
class HocSinh implements Serializable {  
  
    private String ten;  
    private int tuoi;  
  
    public String getTen() {  
        return ten;  
    }  
  
    public void setTen(String ten) {  
        this.ten = ten;  
    }  
  
    public int getTuoi() {  
        return tuoi;  
    }  
  
    public void setTuoi(int tuoi) {  
        this.tuoi = tuoi;  
    }  
}  
  
public class JavaAndroidVn {  
  
    public static void main(String[] args) throws IOException {  
        ArrayList<HocSinh> listHS = new ArrayList(); // Khai báo 1 đối tượng ArrayList, các phần tử tạo ra từ lớp HocSinh  
  
        Scanner input = new Scanner(System.in);  
        System.out.println("Nhập số học sinh: ");  
        int n = input.nextInt();  
  
        //Lấy dữ liệu và ghi vào file  
  
        for (int i = 0; i < n; i++) {  
            HocSinh x = new HocSinh(); //Tạo đối tượng x để lưu tạm thời dữ liệu  
            System.out.println("Học sinh thứ " + i + ": ");  
            input.nextLine(); //Dòng này để tránh bị trượt dòng!  
  
            System.out.print("Tên: ");  
            String tenX = input.nextLine();  
  
            System.out.print("Tuổi: ");  
            int tuoiX = input.nextInt();  
  
            x.setTen(tenX);  
            x.setTuoi(tuoiX);  
  
            listHS.add(x);  
  
        }  
  
  
        try {  
            FileOutputStream f = new FileOutputStream("E:\\hocsinh.dat");  
            ObjectOutputStream oOT = new ObjectOutputStream(f); // Sử dụng để ghi file theo từng Object  
            oOT.writeObject(listHS); // Ghi  Object là đối tượng x xuống file  
        } catch (FileNotFoundException ex) {  
            Logger.getLogger(JavaAndroidVn.class.getName()).log(Level.SEVERE, null, ex);  
        }  
  
  
        //Đọc dữ liệu từ file, lấy các object ra rồi gán vào ListHS  
  
        ArrayList<HocSinh> listHS2 = new ArrayList(); // Khai báo 1 đối tượng ArrayList thứ 2, các phần tử tạo ra từ lớp HocSinh  
        //Dùng để lưu dữ liệu riêng khi ta đọc file!                   
        try {  
            FileInputStream f2 = new FileInputStream("E:\\hocsinh.dat");  
            ObjectInputStream oIT = new ObjectInputStream(f2); // Sử dụng để đọc file theo từng Object  
            listHS2 = (ArrayList<HocSinh>) oIT.readObject(); //Ép kiểu đối tượng lấy từ file ra về dạng ArrayList<HocSinh>  
            oIT.close();  
            f2.close();  
        } catch (IOException io) {  
            System.out.println("Có lỗi xảy ra!");  
        } catch (ClassNotFoundException ex) {  
            System.out.println("Không tìm thấy class");  
        }  
  
        //In thông tin ra ngoài! Thông tin lấy từ đối tượng ListHS2 vừa đọc từ file  
        System.out.println("Thông tin danh sách vừa nhập vào là: ");  
        for (int i = 0; i < listHS2.size(); i++) {  
            System.out.print("Học sinh thứ " + i);  
            System.out.print(" - Tên " + listHS2.get(i).getTen() + " Tuổi: " + listHS2.get(i).getTuoi() + "\n");  
        }  
  
  
    }  
}

## Bài tập 7 (String, StringBuilder)

Bổ sung các bài tập về chuỗi, ghép chuỗi, so sánh chuỗi, …

7.1. Write a Java program to concatenate a given string to the end of another string

1. **public** **class** Exercise7 {
3. **public** **static** **void** main(String[] args)
4. {
5. String str1 = "PHP Exercises and ";
6. String str2 = "Python Exercises";
8. System.out.println("String 1: " + str1);
9. System.out.println("String 2: " + str2);

12. // Concatenate the two strings together.
13. String str3 = str1.concat(str2);
15. // Display the new String.
16. System.out.println("The concatenated string: " + str3);
17. }
18. }

7.2. Write a Java program to check whether two String objects contain the same data

1. **public** **class** Exercise13 {
3. **public** **static** **void** main(String[] args)
4. {
5. String columnist1 = "Stephen Edwin King";
6. String columnist2 = "Walter Winchell";
7. String columnist3 = "Mike Royko";
9. // Are any of the above Strings equal to one another?
10. **boolean** equals1 = columnist1.equals(columnist2);
11. **boolean** equals2 = columnist1.equals(columnist3);
13. // Display the results of the equality checks.
14. System.out.println("\"" + columnist1 + "\" equals \"" +
15. columnist2 + "\"? " + equals1);
16. System.out.println("\"" + columnist1 + "\" equals \"" +
17. columnist3 + "\"? " + equals2);
18. }
19. }

7.3. Write a Java program to compare a given string to another string, ignoring case considerations

1. **public** **class** Exercise14 {
3. **public** **static** **void** main(String[] args)
4. {
5. String columnist1 = "Stephen Edwin King";
6. String columnist2 = "Walter Winchell";
7. String columnist3 = "stephen edwin king";
9. // Test any of the above Strings equal to one another
10. **boolean** equals1 = columnist1.equalsIgnoreCase(columnist2);
11. **boolean** equals2 = columnist1.equalsIgnoreCase(columnist3);
13. // Display the results of the equality checks.
14. System.out.println("\"" + columnist1 + "\" equals \"" +
15. columnist2 + "\"? " + equals1);
16. System.out.println("\"" + columnist1 + "\" equals \"" +
17. columnist3 + "\"? " + equals2);
18. }
19. }

## Bài tập 7.1 (command line)

Xây dựng lớp **Calculation** và xây dựng phương thức main để nhập vào các tham số gồm có toán hạng thứ 1, toán hạng thứ 2, và phép toán của 2 toán hạng đó (+, -, x, /) vào mảng chuỗi của command line để thực hiện tính toán.

# Session 9

## Bài tập 1 (static)

Xây dựng lớp Max, với phương thức static max trả về max 2 số nguyên nhập vào. Sau đó xây dựng lớp TestMax, có phương thức main gọi đến phương thức static max trong lớp Max.

## Bài tập 1.1 (static)

Write a static method max3() that takes three int arguments and returns the value of the largest one. Add an overloaded function that does the same thing with three double values.

public static int max3(int a, int b, int c) {

int max = a;

if (b > max) max = b;

if (c > max) max = c;

return max;

}

public static double max3(double a, double b, double c) {

double max = a;

if (b > max) max = b;

if (c > max) max = c;

return max;

}

## Bài tập 1.2

Write a static method majority() that takes three boolean arguments and returns true if at least two of the arguments are true, and false otherwise. Do not use an if statement.

*Solution*:

|  |
| --- |
| public static boolean majority(boolean a, boolean b, boolean c) {  return (a && b) || (a && c) || (b && c);  } |

## Bài tập 2 (static)

Xây dựng lớp Grade, với phương thức static **printGrade** trả về các ký tự A, B, C, D, F tương ứng với số điểm nhập vào. Sử dụng if-else-if

* Nếu số điểm >=90, in ra A
* Nếu số điểm >=80 và <90, in ra B
* Nếu số điểm >=70 và <80, in ra C
* Nếu số điểm >=60 và <70, in ra D
* Nếu số điểm <60, in ra F.

Xây dựng phương thức main trong lớp TestGrade để gọi đến phương thức printGrade trong lớp Grade, tham số truyền vào là số điểm nhập vào.

## Bài tập 1.2.1

Viết chương trình nhập vào chữ hoa và chuyển đổi nó sang ký tự chữ thường. Xây dựng phương thức static convertUpperCase(char c) trong lớp **CharThings**, sau đó gọi nó từ hàm main

## Bài tập 1.2.2

Viết chương trình nhập vào chữ hoa và chuyển đổi nó sang ký tự chữ thường. Xây dựng phương thức static convertUpperCase(char c) trong lớp **CharThings**, phương thức này trả về ký tự hoa tương ứng với ký tự thường c. Phương thức này sẽ được gọi từ hàm main.

## Bài tập 1.2.3

Tạo 1 lớp Calculator, xây dựng 2 phương thức

* Phương thức static max(int a, int b, int c) trả về giá trị lớn nhất trong 3 số a, b, c trên.
* Phương thức static tính trung bình cộng của 3 số a, b, c: average(int a, int b, int c).
* Trong phương thức main, gọi tới 2 phương thức này để sử dụng, sau khi nhập vào 3 số a, b, c

Write a Java method to find the smallest number among three numbers. [Go to the editor](http://www.w3resource.com/java-exercises/method/index.php#editorr)  
Test Data:  
Input the first number: 25  
Input the Second number: 37  
Input the third number: 29   
Expected Output:

The smallest value is 25.0

Write a Java method to compute the average of three numbers. [Go to the editor](http://www.w3resource.com/java-exercises/method/index.php#editorr)  
Test Data:  
Input the first number: 25   
Input the second number: 45  
Input the third number: 65  
Expected Output:

The average value is 45.0

## Bài tập 1.2.4

Tạo 1 lớp StringManipulation, với các phương thức:

* Phương thức static checkCharacter(String str, char a) trả về true nếu a tồn tại trong chuỗi str, và false nếu ngược lại.
  + *Gợi ý:* 
    - *Sử dụng vòng lặp for để duyệt chuỗi*
    - *Sử dụng str.lenghth() để trả về số ký tự trong chuỗi*
    - *Sử dụng charAt(i) để trả về ký tự nào đó trong chuỗi*
* Phương thức countVowel(String str) trả về số các nguyên âm tồn tại trong chuỗi str.
* Viết phương thức main để gọi phương thức checkCharacter để kiểm tra ký tự trong chuỗi nào đó và countVowel để đếm số nguyên âm có trong chuỗi.

Write a Java method to count all vowels in a string

1. **import** java.util.Scanner;
2. **public** **class** Exercise4 {
3. **public** **static** **void** main(String[] args)
4. {
5. Scanner in = **new** Scanner(System.in);
6. System.out.print("Input the string: ");
7. String str = in.nextLine();
9. System.out.print("Number of  Vowels in the string: " + count\_Vowels(str)+"\n");
10. }
11. **public** **static** **int** count\_Vowels(String str)
12. {
13. **int** count = 0;
14. **for** (**int** i = 0; i < str.length(); i++)
15. {
16. **if** (str.charAt(i) == 'a' || str.charAt(i) == 'e' || str.charAt(i) == 'i'
17. || str.charAt(i) == 'o' || str.charAt(i) == 'u')
18. {
19. count++;
20. }
21. }
22. **return** count;
23. }
24. }

## Bài tập 1.2.5

Tạo 1 lớp Prime, xây dựng phương thức static boolean isPrime(int number) để trả về true, nếu number là số nguyên tố, và trả về false, nếu number không phải là số nguyên tố. Từ main gọi đến phương thức này để kiểm tra.

## Bài tập 1.2.6

Tạo 1 lớp Square, xây dựng phương thức static boolean isSquare(int number) để trả về true, nếu number là số chính phương, và trả về false, nếu number không phải là số chính phương. Từ main gọi đến phương thức này để kiểm tra.

# Session 10

## Bài tập 1/10

Xây dựng 1 lớp **Shape** gồm có các thuộc tính protected sau: color (**String**), filled (**boolean**) cùng các constructors (có tham số và không có tham số) và getters/setters.

Xây dựng lớp **Circle** kế thừa từ lớp **Shape**, có thuộc tính private float bổ sung là **radius**, xây dựng constructor có tham số để khởi tạo các thuộc tính cho đối tượng Circle (ngoài thuộc tính radius, còn có các thuộc tính được kế thừa từ Shape), xây dựng phương thức bổ sung **showInfo()** để hiển thị các thông tin các thuộc tính của đối tượng **Circle**.

Xây dựng lớp **TestShape** có phương thức **main** để khởi tạo đối tượng **Circle,** thiết lập các thuộc tính và in ra các thông tin của nó.

## Bài tập 1.1 (override)

Tạo 1 lớp Circle, với thuộc tính protected double radius, 2 constructor có và không có tham số để khởi tạo thuộc tính radius, phương thức getRadius() để lấy giá trị thuộc tính radius ra, phương thức getArea() để trả về giá trị diện tích của đối tượng Circle~~, phương thức toString() để in ra thông tin đối tượng Circle~~.

Tạo lớp Cylinder (hình trụ) kế thừa từ lớp Cirlce, có thuộc tính bổ sung private double là height, tạo 2 constructor có và không có tham số để khởi tạo thuộc tính radius (được kế thừa) và height cho đối tượng Cylinder, viết thêm các phương thức bổ sung getHeight() để trả về chiều cao của đối tượng Cylinder, override lại phương thức tính diện tích getArea() để tính diện tích Cylinder, ~~và cũng override lại phương thức in thông tin của đối tượng Cylinder toString()~~

Tạo lớp TestMain, có phương thức main để khởi tạo đối tượng Circle và Cylinder. Khởi tạo các thuộc tính bằng các constructors, thiết lập các thuộc tính, sau đó in ra diện tích và thông tin của mỗi đối tượng. ~~Ta có thể lưu ý tại đây về upcast và downcast và sự cho phép gọi phương thức căn cứ vào kiểu tham chiếu được khai báo, chứ ko phải đối tượng được trỏ tới.~~

// The superclass Circle

public class Circle {

// private instance variable

private double radius;

// Constructor

public Circle(double radius) {

this.radius = radius;

}

// Getter

public double getRadius() {

return this.radius;

}

// Return the area of this circle

public double getArea() {

return radius \* radius \* Math.PI;

}

// Describe itself

public String toString() {

return "Circle[radius=" + radius + "]";

}

}

// The subclass Cylinder

public class **Cylinder extends Circle** {

// private instance variable

private double height;

// Constructor

public Cylinder(double height, double radius) {

super(radius); // khởi tạo đầy đủ chứ ko sử dụng super cho bài tập này

this.height = height;

}

// Getter

public double getHeight() {

return this.height;

}

// Return the volume of this cylinder

public double getVolumne() {

return super.getArea() \* height;

}

// Override the inherited method to return the surface area

@Override

public double getArea() {

return 2.0 \* Math.PI \* getRadius() \* height;

}

// Override the inherited method to describe itself

@Override

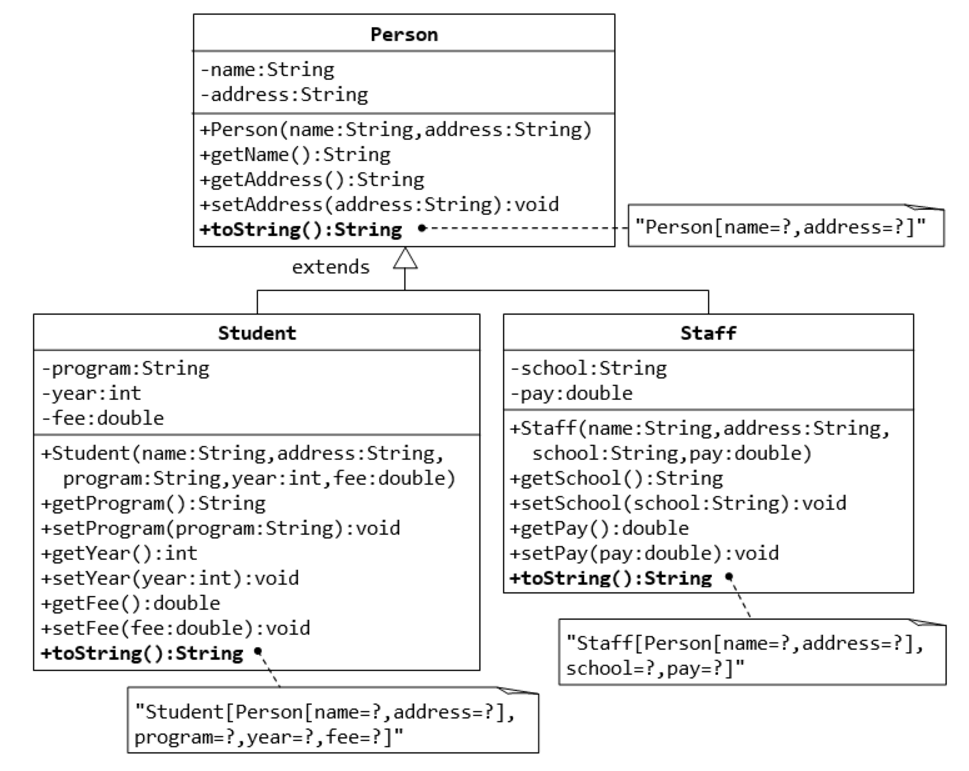
public String toString() {

return "Cylinder[height=" + height + "," + super.toString() + "]";

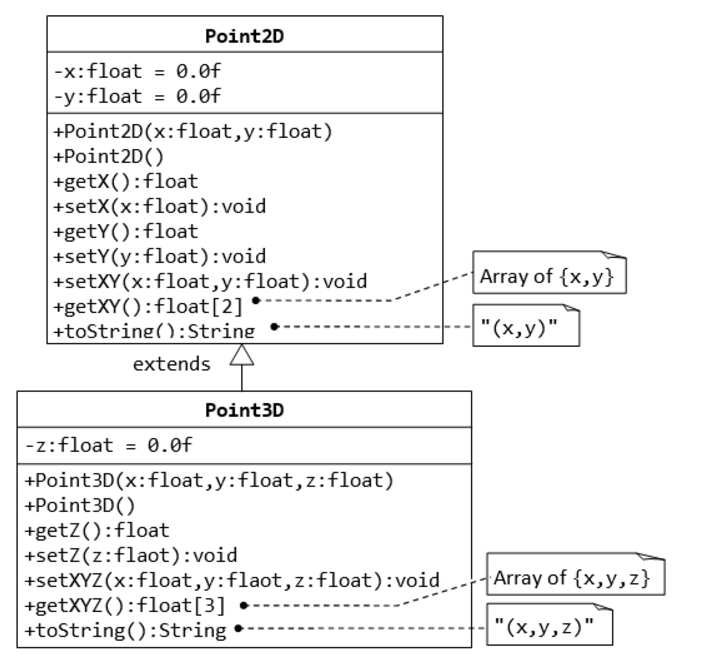
}

}

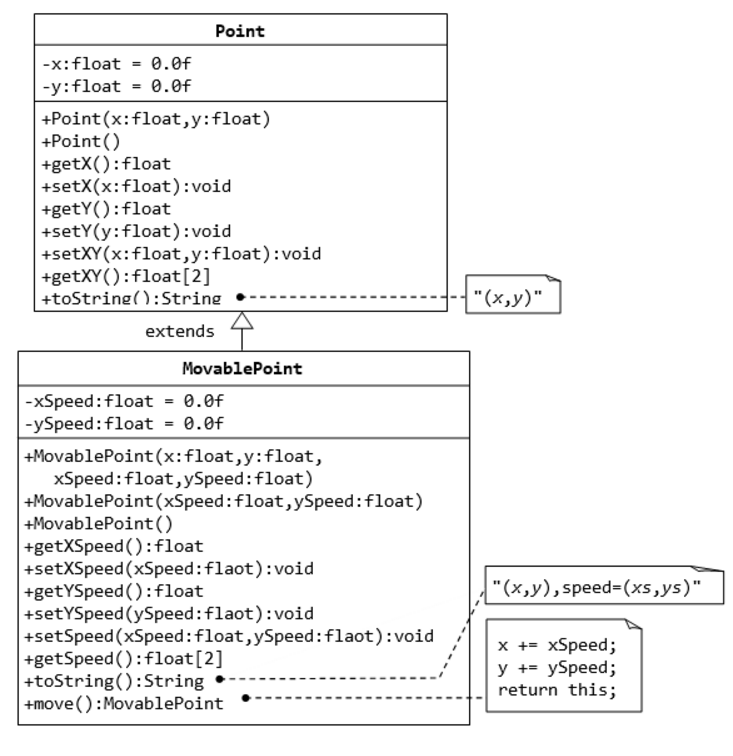
## Bài tập 1.2 (override)



## Bài tập 1.3 (override)



## Bài tập 1.4



## Bài tập 2 (override, polymorphism)

Xây dựng 1 lớp **Shape** gồm có các thuộc tính **protected** sau: color (**String**), filled (**boolean**), phương thức khởi tạo cho **color**, **filled** cùng các getters và setters, phương thức **getArea()** luôn trả về 0.

Xây dựng lớp **Circle** kế thừa từ lớp **Shape**, có thuộc tính bổ sung là **radius**, xây dựng hàm khởi tạo có tham số, override lại phương thức **getArea()** để trả về giá trị diện tích của **Circle**.

Xây dựng lớp **TestShape** có phương thức main để khởi tạo đối tượng **Circle** và in ra thông tin diện tích của nó.

## Bài tập 2.1 (override, polymorphism)

/\*

\* The superclass Monster defines the expected common behaviors for its subclasses.

\*/

public class Monster {

// private instance variable

private String name;

// Constructor

public Monster(String name) {

this.name = name;

}

// Define common behavior for all its subclasses

public String attack() {

return "!^\_&^$@+%$\* I don't know how to attack!";

// We have a problem here!

// We need to return a String; else, compilation error!

}

}

public class FireMonster extends Monster {

// Constructor

public FireMonster(String name) {

super(name);

}

// Subclass provides actual implementation

@Override public String attack() {

return "Attack with fire!";

}

}

public class WaterMonster extends Monster {

// Constructor

public WaterMonster(String name) {

super(name);

}

// Subclass provides actual implementation

@Override public String attack() {

return "Attack with water!";

}

}

public class StoneMonster extends Monster {

// Constructor

public StoneMonster(String name) {

super(name);

}

// Subclass provides actual implementation

@Override public String attack() {

return "Attack with stones!";

}

}

public class **TestMonster** {

public static void main(String[] args) {

// Program at the "interface" defined in the superclass.

// Declare instances of the superclass, substituted by subclasses.

Monster m1 = new FireMonster("r2u2"); // upcast

Monster m2 = new WaterMonster("u2r2"); // upcast

Monster m3 = new StoneMonster("r2r2"); // upcast

// Invoke the actual implementation

System.out.println(m1.attack()); // Run FireMonster's attack()

System.out.println(m2.attack()); // Run WaterMonster's attack()

System.out.println(m3.attack()); // Run StoneMonster's attack()

// m1 dies, generate a new instance and re-assign to m1.

m1 = new StoneMonster("a2b2"); // upcast

System.out.println(m1.attack()); // Run StoneMonster's attack()

// We have a problem here!!!

Monster m4 = new Monster("u2u2");

System.out.println(m4.attack()); // garbage!!!

}

}

## Bài tập 3/40 (abstract, override, polymorphism)

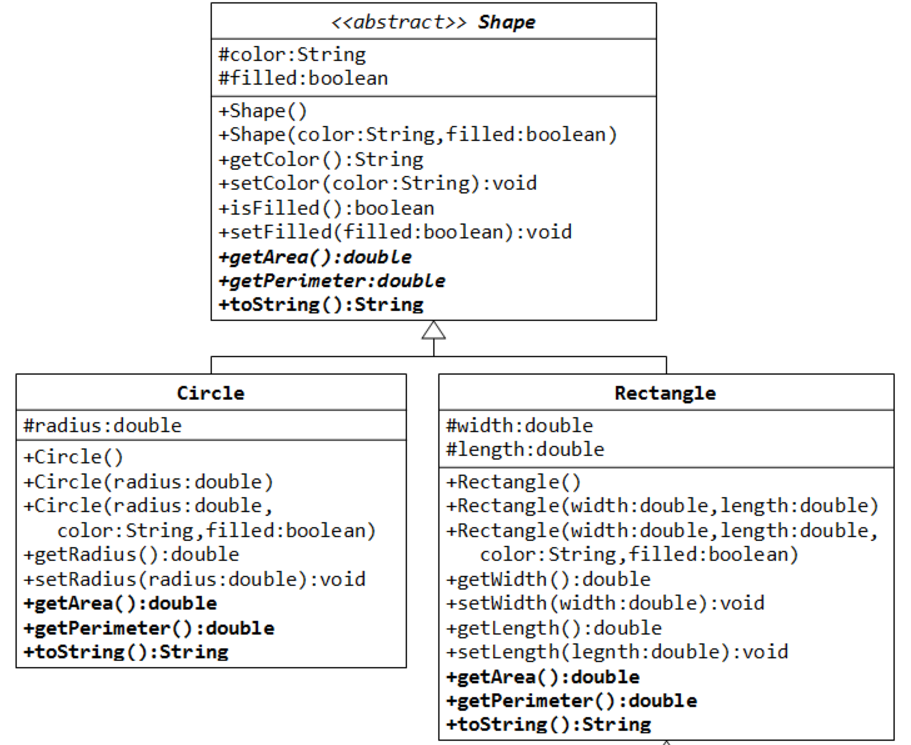
Xây dựng 1 lớp **Shape** gồm có các thuộc tính **protected** sau: color (**String**), filled (**boolean**), phương thức khởi tạo cho color, filled và phương thức **abstract** **getArea()**.

Xây dựng lớp **Circle**, kế thừa từ lớp **Shape**, có thuộc tính bổ sung là **radius**, xây dựng phương thức khởi tạo, override lại phương thức **getArea()** để trả về giá trị diện tích của **Circle**.

Xây dựng lớp **Rectangle**, kế thừa từ lớp **Shape**, có thuộc tính bổ sung là **width**, và **height**, xây dựng phương thức khởi tạo, override lại phương thức **getArea()** để trả về giá trị diện tích của **Rectangle**.

Xây dựng lớp **TestShape** có phương thức **main** để khởi tạo đối tượng **Circle** và đối tượng **Rectangle**. Sau đó in ra diện tích của từng đối tượng.

## Bài tập 4 (abstract, override, polymorphism)



# Session 11

## Bài tập 1

Xây dựng **interface** **Shape**, có phương thức **getArea()**

Xây dựng lớp **Circle**, **Rectangle** implement interface **Shape**, trong mỗi lớp định nghĩa lại method **getArea()** cho phù hợp với mỗi lớp. Trong mỗi lớp, cũng định nghĩa các trường phù hợp, constructors, setters và getters.

Xây dựng lớp **TestShape** có phương thức main để khởi tạo đối tượng **Circle** và **Rectangle**. Thiết lập các giá trị thuộc tính cho mỗi đối tượng. Sau đó in ra diện tích của từng đối tượng.

## Bài tập 2

Tạo interface Movable với các phương thức moveUp(), moveDown(), moveLeft(), moveRight(). Sau đó tạo lớp MovablePoint implements interface Movable. Lớp MovablePoint có 2 thuộc tính private là int x, int y. Trong lớp này, cứ tạo ra các constructor và getter/setters và override các phương thức được khai báo trong interface.

Tạo lớp Test để khởi tạo đối tượng MovablePoint, sử dụng cách thông thường, và khai báo sử dụng kiểu upcast.

/\*

\* The Movable interface defines a list of public abstract methods

\* to be implemented by its subclasses

\*/

public interface **Movable** { // use keyword "interface" (instead of "class") to define an interface

// An interface defines a list of public abstract methods to be implemented by the subclasses

public void moveUp(); // "public" and "abstract" optional

public void moveDown();

public void moveLeft();

public void moveRight();

}

// The subclass MovablePoint needs to implement all the abstract methods

// defined in the interface Movable

public class **MovablePoint implements Movable** {

// Private member variables

private int x, y; // (x, y) coordinates of the point

// Constructor

public MovablePoint(int x, int y) {

this.x = x;

this.y = y;

}

@Override

public String toString() {

return "(" + x + "," + y + ")";

}

// Need to implement all the abstract methods defined in the interface Movable

@Override

public void moveUp() {

y--;

}

@Override

public void moveDown() {

y++;

}

@Override

public void moveLeft() {

x--;

}

@Override

public void moveRight() {

x++;

}

}

public class **TestMovable** {

public static void main(String[] args) {

MovablePoint p1 = new MovablePoint(1, 2); // upcast

System.out.println(p1);

p1.moveDown();

System.out.println(p1);

p1.moveRight();

System.out.println(p1);

// Test Polymorphism

Movable p2 = new MovablePoint(3, 4); // upcast

p2.moveUp();

System.out.println(p2);

MovablePoint p3 = (MovablePoint)p2; // downcast

System.out.println(p3);

}

}

# Session 12

## Bài tập 1.3

public class Main {  
      
    public static void main(String[] args) {  
  
        **// Enter the following values and see what exception is caught.  
        //    0, 4, character, <no value>**  
        try {  
  
            String value = JOptionPane.showInputDialog(null, "Enter value:");  
              
           **// Non-numerica value will result an NumberFormatException**  
            int divisor = Integer.parseInt(value);  
              
           **// If the divisor is 0, it will result in ArithmetricException**  
            System.out.println(3/divisor);  
  
        **} catch (NumberFormatException nfe){  
            System.out.println("Exception caught by this program: Enter numeric value.");  
        } catch (ArithmeticException exc) {  
            System.out.println("Exception caught by this program: Divisor was 0.");  
        }**  
        System.out.println("After exception.");  
    }  
      
}

## Bài tập 1

Viết chương trình java, nhập vào các phần tử.  
  
-Hiện ra số phần tử nhập vào  
-Trung bình cộng của các số  
-Đưa ra các phần tử không phải là số(dùng Exception)  
  
Output mẫu như sau:  
  
java Main 4 3 5c d 30 bb 23  
Number of elements = 7  
Average = 15  
Not a number: 5c, d, bb

## Bài tập 1.1\_534

Viết chương trình nhập số nguyên và hiển thị số nguyên. Nếu nhập sai định dạng thì cho nhập lại, đến khi nào nhập đúng rồi hiển thị ra và kết thúc chương trình

*Gợi ý:*

* *InputMismatchException ex*

## Bài tập 1.2

class SqRoot2 {

public static void main(String[] args) {

try {

double d = getInput(args[0]);

double root = Math.sqrt(d);

System.out.println(

"The square root of " + d + " is " + root);

}

catch (NumberFormatException e) {

System.out.println("Be sure to enter a number.");

}

catch (ArrayIndexOutOfBoundsException e) {

System.out.println("Enter number as first parameter.");

}

catch (ImNumberException e) {

System.out.println("Result will be imaginary number.");

}

}

static double getInput(String s) throws ImNumberException {

double d = new Double(s).doubleValue();

// Throw an ImNumberException if d is less than 0

return d;

}

}

## Bài tập 2

/\*

This program will compute and print one of the solutions

to an equation of the form A\*X\*X + B\*X + C = 0, where

A, B, and C are numbers entered by the user.

\*/

public class Quadratic {

public static void main(String[] args) {

double A, B, C; // Coefficients in the equation.

double solution; // The solution computed for the equation.

boolean goAgain; // Set to true if the user wants to

// solve another equation.

TextIO.putln("This program will print a solution of an equation");

TextIO.putln("of the form A\*X\*X + B\*X + C = 0, where A, B, and");

TextIO.putln("C are values that you enter.");

do {

/\* Get the coefficients from the user. \*/

TextIO.putln();

TextIO.putln("Enter values for A, B, and C:");

TextIO.put("A = ");

A = TextIO.getlnDouble();

TextIO.put("B = ");

B = TextIO.getlnDouble();

TextIO.put("C = ");

C = TextIO.getlnDouble();

TextIO.putln();

/\* Print the solution, or an error message, if

there is no solution. \*/

try {

solution = root(A,B,C);

TextIO.putln("A solution of the equation is " + solution);

}

catch (IllegalArgumentException e) {

TextIO.putln("Sorry, I can't find a solution.");

TextIO.putln(e.getMessage());

}

/\* Find out whether the user wants to go again. \*/

TextIO.putln();

TextIO.put("Do you want to solve another equation? ");

goAgain = TextIO.getlnBoolean();

} while (goAgain);

} // end main

static double root(double A, double B, double C)

throws IllegalArgumentException {

// Returns the larger of the two roots of

// the quadratic equation A\*x\*x + B\*x + C = 0.

// (Throws an exception if A == 0 or B\*B-4\*A\*C < 0.)

if (A == 0) {

throw new IllegalArgumentException("A can't be zero.");

}

else {

double disc = B\*B - 4\*A\*C;

if (disc < 0)

throw new IllegalArgumentException("Discriminant < zero.");

return (-B + Math.sqrt(disc)) / (2\*A);

}

} // end root()

} // end class