    LocalDate D2 = LocalDate.parse("2019-01-01", DateTimeFormatter.ISO\_DATE);

    LocalTime d = LocalTime.of(1, 15, 30);

    d.withHour(2);

LocalTime 不可更改

-cp , -classpath, --class-path

Javac -g generate all debug information

    short a = 'A';

    char c ="A";

    foat f =1.0;

    double dd =1;

string cannot convert to char

double cannot convert to float

short == char

In interface if there is a static method it must has a body.

Method signature = number of argument, type of argument and order of argument

Overload : different argument and same method name

Override : same method name and same argument and subclass return type

对于override的函数, 或者父子都是static，或者父子都non-static. Static不能只出现的父类或者只出现在子类中。 父类方法的accesible要高于子类的accesible.

JPMS

Export package and require module

All public classes are exported

Switch only workd with short , char, int ,byte.

Throws for checked exception

Super() this() constructor call must be the first statement in a constructor . 构造函数里调用super()一定放在最前面。

(long)1.8 result is 1

Switch must has a default or it cannot cover all situations. Switch type cannot be bool only primative type. Switch 一定要有break和default

Var 初始化是 int

var can be used for declaring local variables, including index variables of for-loops and resource variables of the try-with-resources statement.

var cannot be used for fields, method parameters, and method return types.

Unreach statement 会导致编译错误

从接口那继承来的方法一定要加上public ,因为接口的抽象方法都是public

String的concat方法两个被连接的字符串都不能为null，否则nullpointexception

如果两个null字符串用+， 可以输出nullnull

Stringbuilder初始化capacity是16

Field initialization 在block initialization之后， construct initialization之前。

一个类从抽象类那里继承后，要完成所有的抽象方法，否则就自己声明为抽象类，原来的抽象方法还是抽象方法。

Jdeps –generate-module-info

与array相关的方法有binarysearch, copy, equal, fill, sort, hashcode

与list相关的方法有add,remove, get, set, contain, iterator, toArray

同样都是import进来的方法，explicity 的优先级高于\*

String.replace(A,B) A,B字符字符串都可以

Byte < short < int < long < Integer

抽象类的抽象方法一定加abtract, 接口的方法可以不加

   List<String> strings = Arrays.asList("A","B");

         strings.remove("A");

strings是不可改变的

         List<String> strings = new ArrayList<>();

         strings.add("A");

         strings.add("B");

strings是可以改变的

         List<String> strings = new ArrayList<>(List.of("A","B"));

         strings.add("A");

         strings.add("B");

         System.out.println(strings);

         strings.remove("A");

strings是可以改变的

         List<String> strings = List.of("A","B");

         strings.add("A");

         strings.add("B");

         strings.remove("A");

strings是不可改变的

1. default array 是固定的数组，原始类型，或者引用类型的数组

        String str[] = new String[5];

        for (String s : str)

            System.out.print(s + " ");

null null null null null

       ArrayDemo ademo[] = new ArrayDemo[5];

        for (ArrayDemo val : ademo)

            System.out.print(val + " ");

null null null null null

1. jagged array是不固定的数组

int arr[][] = new int[2][];

arr[0] = new int[3];

arr[1] = new int[2];

int count = 0;

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

     for(int j=0; j<arr[i].length; j++)

        arr[i][j] = count++;

1. 然后会出现数组越界的问题

        int ar[] = {1, 2, 3, 4, 5};

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

          System.out.println(ar[i]);

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 5

1. Arraylist是动态数组

        // Create an arrayList with initial capacity 2

        ArrayList<Integer> arrL = new ArrayList<Integer>(2);

        // Add elements to ArrayList

        arrL.add(1);

        arrL.add(2);

// Access elements of ArrayList

        System.out.println(arrL.get(0));

Output:

1

        ArrayList<Integer> arrL = new ArrayList<Integer>(2);

        // Add elements to ArrayList

        arrL.add(1);

        arrL.add(2);

        arrL.add(3);

Output:

[1, 2, 3]

Array can contain both primitive data types as well as objects of a class depending on the definition of the array. However, ArrayList only supports object entries, not the primitive data types.

       int[] array = new int[3];

        // allowed, however, need to be intialized

        Test[] array1 = new Test[3];

        // not allowed (Uncommenting below line causes

        // compiler error)

        // ArrayList<char> arrL = new ArrayList<char>();

        // Allowed

        ArrayList<Integer> arrL1 = new ArrayList<>();

        ArrayList<String> arrL2 = new ArrayList<>();

        ArrayList<Object> arrL3 = new ArrayList<>();

1. Arraylist can convert to array

Syntax:

**public Object[] toArray()**

       List<Integer> al = new ArrayList<Integer>();

        al.add(10);

        al.add(20);

        al.add(30);

        al.add(40);

        Object[] objects = al.toArray();

        // Printing array of objects

        for (Object obj : objects)

            System.out.print(obj + " ");

Output:

10 20 30 40

或者

**public T[] toArray(T[] arr)**

        List<Integer> al = new ArrayList<Integer>();

        al.add(10);

        al.add(20);

        al.add(30);

        al.add(40);

        Integer[] arr = new Integer[al.size()];

        arr = al.toArray(arr);

        for (Integer x : arr)

            System.out.print(x + " ");

Output:

10 20 30 40

1. Using Iterator, you can traverse List, Set and Queue type of objects. But using ListIterator, you can traverse only List objects.
2. List list = new ArrayList();
4. list.add("ONE");
6. list.add("TWO");
8. list.add("THREE");
10. //Traversing list elements using Iterator
11. Iterator iterator1 = list.iterator();
13. while (iterator1.hasNext())
14. {
15. System.out.println(iterator1.next());
16. }

**This, super 都不能用在static 方法中**

List copyof 是不可变数组

    String s1 = "abcd";

    StringBuilder s2 = new StringBuilder("abcd");

    StringBuilder s3 = new StringBuilder("abcd");

    System.out.println(s2.equals(s3));

    System.out.println(s1.equals(s2.toString()));

Output

false

true

    String s1 = "abcd";

    StringBuilder s2 = new StringBuilder("abcd");

    StringBuilder s3 = new StringBuilder("abcd");

    System.out.println(s2.toString().equals(s3.toString()));

    System.out.println(s1.equals(s2.toString()));

Output

True

True