1. for语法格式：

for(表达式1; 表达式2; 表达式3)

for语句是先判断后执行，先为循环控制变量赋初值，再判断循环条件是否满足，满足则执行循环体，不满足，循环结束

while语法格式：

while(布尔表达式)

语句；

While语句先判断后执行，满足循环条件执行循环体

do-while语法格式：

do{

语句；

}while(布尔表达式)

do-while语句先执行后判断，先执行循环体再判断布尔表达式的值，值为true，循环继续，否则循环结束 执行下一条，循环至少要执行一次

1. java定义了三种关键字表示四种访问权限。

Public表示公有权限：声明公共成员，该成员被所有类访问

Protected声明保护成员，该成员能被当前类及其子类或当前包中的其他类访问。

Private声明私有成员，该成员仅能被当前类成员访问

无修饰符表示缺省权限，该成员能被当前类及当前包中其他类访问

3. 用位移运算。2 << 3

因为将一个数左移n位，就相当于乘以了2的n次方，那么，一个数乘以8只要将其左移3位即可，而计算机底层对数字的运算，实际上就是二进制运算，所以效率最高，所以，2乘以8等於几的最效率的方法是2 << 3。

4．对于short s1 = 1; s1 = s1 + 1;由于s1+1运算时会自动提升表达式的类型，所以结果是int型，再赋值给short类型s1时，编译器将报告需要强制转换类型的错误。对于short s1 = 1; s1 += 1;由于 +=是java语言规定的运算符，java编译器会对它进行特殊处理，因此可以正确编译。

5.

\* 抽象类可以有构造方法，接口中不能有构造方法。

\* 抽象类中可以有普通成员变量，接口中没有普通成员变量

\* 抽象类中可以包含非抽象的普通方法，接口中的所有方法必须都是抽象的，不能有非抽象的普通方法。

\* 抽象类中的抽象方法的访问类型可以是public，protected和（默认类型,虽然

eclipse下不报错，但应该也不行），但接口中的抽象方法只能是public类型的，并且默认即为public abstract类型。

\* 抽象类中可以包含静态方法，接口中不能包含静态方法

\* 抽象类和接口中都可以包含静态成员变量，抽象类中的静态成员变量的访问类型可以任意，但接口中定义的变量只能是public static final类型，并且默认即为public static final类型。

\* 一个类可以实现多个接口，但只能继承一个抽象类。

6.成员方法使用this引用来调用该方法当前对象

\*\* this用于指代调用成员方法的当前对象自身

\*\* 访问本类的属性和成员方法

\*\* 调用本类重载的构造方法

7. 设计模式有三种：单态模式、外观模式、简单工厂模式、

观察者模式、装饰模式

8. 泛型是对Java语言的数据类型系统的一种扩展，以支持创建可以按类型进行参数化的类。可以把类型参数看做是使用参数化类型时指定的类型的一个占位符。

优点：提高Java程序的类型安全；消除强制类型转换；提高代码的重用率。 泛型List可以实例化为只能存储某种特定类型的数据，普通List可以实例化为存储各种类型的数据。通过使用泛型List对象，可以规范集合对象中存储的数据类型，在获取集合元素时不用进行任何强制类型转换。

9. public class Ex15 { public static void main(String[] args) {

HashMap map=new HashMap();

map.put(“张三“,90);

map.put(“李四“,83);

System.out.println(“修改前的成绩：“); System.out.println(map); map.put(“李四“,100); System.out.println(“修改后的成绩：“); System.out.println(map); } }

10. 异常是指程序运行过程中产生的错误，它出现在程序运行过程中。 异常处理机制是为程序提供错误处理的能力。根据这个机制，对程序运行时可能遇到的异常情况，预先提供一些处理的方法。在程序执行代码的时候，一旦发生异常，程序会根据预定的处理方法对异常进行处理，处理完成后，程序进行运行。

11. 四大组件： Activity service ContentProvider BroadCastReceiver

Activity：Activity是Android程序与用户交互的窗口，是Android构造块中最基本的一种，它需要为保持各界面的状态，做很多持久化的事情，妥善管理生命周期以及一些跳转逻辑  
service：后台服务于Activity，封装有一个完整的功能逻辑实现，接受上层指令，完成相关的食物，定义好需要接受的Intent提供同步和异步的接口

Content Provider：是Android提供的第三方应用数据的访问方案，可以派生Content Provider类，对外提供数据，可以像数据库一样进行选择排序，屏蔽内部数据的存储细节，向外提供统一的借口模型，大大简化上层应用，对数据的整合提供了更方便的途径

BroadCast Receiver：接受一种或者多种Intent作触发事件，接受相关消息，做一些简单处理，转换成一条Notification，统一了Android的事件广播模型

12. 常用布局有五种：LinerLayout(线性布局)

RelativeLayout (相对布局) FrameLayout (框架布局) TableLayout (表格布局) AbsoluteLayout(绝对布局)

13. XML解析主要有三种方式，SAX、DOM、PULL。

SAX读取是单向的，优点:不占内存空间、解析属性方便，但缺点就是对于套嵌多个分支来说处理不是很方便。

DOM方式会把整个XML文件加载到内存中去，

PULL常常用在J2ME对于节点处理比较好，类似SAX方式，同样很节省内存，在J2ME中我们经常使用的KXML库来解析。

14.

\*\* 如果自定义适配器，那么在getView方法中要考虑方法传进来的参数contentView是否为null，如果为null就创建contentView并返回，如果不为null则直接使用。在这个方法中尽可能少创建view。\*\* 给contentView设置tag（setTag（）），传入一个viewHolder对象，用于缓存要显示的数据，可以达到图像数据异步加载的效果。  
\*\* 如果listview需要显示的item很多，就要考虑分页加载。比如一共要显示100条或者更多的时候，我们可以考虑先加载20条，等用户拉到列表底部的时候再去加载接下来的20条。

15. SharedPreferences存储数据； 文件存储数据； SQLite数据库存储数据； 使用ContentProvider存储数据； 网络存储数据；

16. 必调用的三个方法：onCreate() --> onStart() --> onResume()，用AAA表示

（1）父Activity启动子Activity，子Actvity退出，父Activity调用顺序如下  
AAA --> onFreeze() --> onPause() --> onStop() --> onRestart() --> onStart(),onResume() …  
（2）用户点击Home，Actvity调用顺序如下  
AAA --> onFreeze() --> onPause() --> onStop() -- Maybe --> onDestroy() – Maybe  
（3）调用finish()， Activity调用顺序如下  
AAA --> onPause() --> onStop() --> onDestroy()  
（4）在Activity上显示dialog， Activity调用顺序如下  
AAA  
（5）在父Activity上显示透明的或非全屏的activity，Activity调用顺序如下  
AAA --> onFreeze() --> onPause()  
（6）设备进入睡眠状态，Activity调用顺序如下  
AAA --> onFreeze() --> onPause()

17. onResume跳转到onPause;onCreat,onStart, onResume

18.

当一个程序第一次启动时，Android会同时启动一个对应的主线程（Main Thread），主线程主要负责处理与UI相关的事件，所以主线程通常又被叫做UI线程。

多线程的实现

1)Activity.runOnUiThread(Runnable)    
2)View.post(Runnable) ;View.postDelay(Runnable , long)    
3)Handler    
4)AsyncTask

19. 这个生命周期跟清单文件里的配置有关系

1、不设置Activity的android:configChanges时，切屏会重新调用各个生命周期

默认首先销毁当前activity,然后重新加载

2、设置Activity的android:configChanges="orientation|keyboardHidden|screenSize"时，切屏不会重新调用各个生命周期，只会执行onConfigurationChanged方法

20. connection.connect()

21. file:///android\_asset/html/hello.html

22. onPause()

23．private class MyHandler extends Handler {

TextView mText；

@Override

public void handleMessage(Message msg) {

super.handleMessage(msg);

switch(msg.what){

case UPDATE ://在收到消息时，对界面进行更新 mText.setText("This update by message");

break; } } }

24. startService()方法和stopService()方法

25. intent可以用来解耦，利用intent可以启动activity，service，broadcast。intent就像是Handler发了一消息，然后系统根据消息器做某些事情。 Handler的出现就是为了解决在主线程做耗时操作。26. drawable layout 横屏竖屏 使用9patch图片， <supports-screens>

1，drawable xhdip，xxhdip里面放一些比较重要的图片，

2，在values dimens.xml里面，配置不同的dimen，对于不同分辨率的屏幕适配不同的dimen ，一般不用。

3，使用9patch图片，里面不能编辑，四周可以。拉伸水平或者竖直方向的两个点所对应的像 素。四个角上是透明的像素，并且可以等比例放大。

27. 存放，节省流量；滑动的时候，不加载图片，停止滑动的时候加载可见的item的图片，节省流量； 服务器更换图片，地址也变，SD卡就会同步的变化。

28. onPreExcute：在ui线程执行一些准备的操作。比如搞一个进度条 doinBackGround：执行耗时的操作。如访问网络，或者调用publishProgress更新进度 onPostExecute：ui线程，执行完成后更新ui 此类用于短时间操作，是Thread和Handler的一个helper。长时间操作使用： Executor, ThreadPoolExecutor and FutureTask 可以有序的运行execute，也可以并行的运行executeOnExecutor 。并行的从线程池中运行多任务，不能保证有序，不建议使用。

29. 运行时异常：我们不用管，会抛给jvm处理，比如空指针异常， 系统会把异常一直往上层抛，一直遇到处理代码。如果没有处理块，到最上层，如果是多线程就由Thread.run()抛出， 如果是单线程就被main()抛出。抛出之后，如果是线程，这个线程也就退出了。如果是主程序抛出的异常，那么这整个程序也就退出了

编译期间异常：try catch finally. 如果不想终止，则必须扑捉所有的运行时异常，决不让这个处理线程退出.

30. 默认情况,如果没有显示的指定service所运行的进程, Service和activity是运行在当前app所在进程的main thread(UI主线程)里面 service里面不能执行耗时的操作(网络请求,拷贝数据库,大文件 ) 在子线程中执行 new Thread(){}.start(); 特殊情况 ,可以在清单文件配置 service 执行所在的进程 , 让service在另外的进程中执行 <service android:process="cn.itcast.xxx"></service>

31. 基本数据类型可以通过. Intent 传递数据

extras.putDouble(key, value) intent.putExtra(name, value) // 通过intent putExtra 方法 基本数据类型 都传递 Bundle bundle = new Bundle(); bundle.putShort(key, value); intent.putExtras(bundle);

intent.putExtras(bundle)

32.

比方说在激活一个新的activity时候, 给intent设置flag Intent的flag添加FLAG\_ACTIVITY\_NEW\_TASK singleinstance 单独的任务栈 这个被激活的activity就会在新的task栈里面„ Intent intent = new Intent(A.this,B.class); intent.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK); startActivity(intent);

33. 一般的基本数据类型 Intent .putextra() intent.getStringextra();

数据的uri, intent.setData() intent.getData();

34. 两种，一种是Tween动画、还有一种是Frame动画。 Tween动画，这种实现方式可以使视图组件移动、放大、缩小以及产生透明度的变化； 可以通过布局文件,可以通过代码

控制View的动画 a) alpha(AlphaAnimation) 渐变透明 b) scale(ScaleAnimation) 渐变尺寸伸缩 c) translate(TranslateAnimation) 画面转换、位置移动 d) rotate(RotateAnimation) 画面转移，旋转动画

35． MVC英文即Model-View-Controller，即把一个应用的输入、处理、输出流程按照Model、View、Controller的方式进行分离，这样一个应用被分成三个层——模型层、视图层、控制层。

Android中界面部分也采用了当前比较流行的MVC框架，在Android中M就是应用程序中二进制的数据，V就是用户的界面。Android的界面直接采用XML文件保存的，界面开发变的很方便。在Android中C也是很简单的，一个Activity可以有多个界面，只需要将视图的ID传递到setContentView()，就指定了以哪个视图模型显示数据。

在Android SDK中的数据绑定，也都是采用了与MVC框架类似的方法来显示数据。在控制层上将数据按照视图模型的要求（也就是Android SDK中的Adapter）封装就可以直接在视图模型上显示了，从而实现了数据绑定。比如显示Cursor中所有数据的ListActivity，其视图层就是一个ListView，将数据封装为ListAdapter，并传递给ListView，数据就在ListView中显示

36、逻辑错误

1.断点 debug   2. logcat ,

界面布局,显示 hierarchyviewer.bat

37. 1、不设置Activity的android:configChanges时，切屏会重新调用各个生命周期，切横屏时会执行一次，切竖屏时会执行两次

2、设置Activity的android:configChanges="orientation"时，切屏还是会重新调用各个生命周期，切横、竖屏时只会执行一次

3、设置Activity的android:configChanges="orientation|keyboardHidden"时，切屏不会重新调用各个生命周期，只会执行onConfigurationChanged方法

38． onCreateOptionsMenu() onOptionsItemSelected()

39.

andriod提供了 Handler 和 Looper 来满足线程间的通信。Handler 先进先出原则。Looper类用来管理特定线程内对象之间的消息交换(Message Exchange)。

 1)Looper: 一个线程可以产生一个Looper对象，由它来管理此线程里的Message Queue(消息队列)。

2)Handler: 你可以构造Handler对象来与Looper沟通，以便push新消息到Message Queue里;或者接收Looper从Message Queue取出)所送来的消息。

 3) Message Queue(消息队列):用来存放线程放入的消息。

 4)线程：UI thread 通常就是main thread，而Android启动程序时会替它建立一个Message Queue。

40. Intent 可以传递的数据类型非常的丰富,java 的基本数据类型和 String 以及他们的数组形式都可以,除此之 外还可以传递实现了 Serializable 和 Parcelable 接口的对象。

41. 12 ,-11

42. false

43. onStart()

44. 可以实现界面间的切换，可以包含动作和动作数据，连接四大组件的纽带

45. getWriteableDatabase()

getReadableDatabase()

46. <receiver android:name=*"NewBroad"*>

<intent-filter>

<action

android:name=*"android.provider.action.NewBroad"*/>

<action>

</intent-filter>

</receiver>

47. good and gbc

48.

主要放置多媒体等数据文件

49. onDestroy()

50. SALES表中qty列最小值大于等于60