网上收集，部分未注明出处，侵删

Java面试 https://blog.csdn.net/qhj1610/article/details/77460543

**1.请列举一些你知道的 jQuery 方法**

1.读写节点的 html 内容obj.html()/obj.html("<p>写节点</p>")

2.读写节点的文本内容obj.text()/obj.text("写文本")

3.读写节点的 value 属性值obj.val()/obj.val("写属性值")

4.读写节点的属性值obj.attr（"属性名"）/obj.val("属性名"，"属性值")

**2.请列举一些常见的Java异常**

**3.请列举一些你知道的SQL关键字**

CREATE DROP ALTER

TABLE VIEW INDEX

SEQUENCE SELECT AS

ORDER BY GROUP BY MODIFY

**4.请列举一些你知道的Linux命令**

关闭系统立即关机：$shutdown -h now

关闭系统10分钟后关机：$shutdown -h +10

重启：$shutdown -r now

在当前目录下创建一个名为 temp 的文件夹：$mkdir temp

查看当前路径：pwd

查看所有网络接口及其状态：$ifconfig -a

使用 up 启动某个接口：$ifconfig eth0 up

使用 down 命令停止某个接口：$ifconfig eth0 down

列出目录下的所有文件：ls -a

**5.javascript 中，使用 typeof 判断一个变量的类 型，可能有哪些返回结果**

typeof 函数查询当前数据类型，返回值可能有：String、Number、boolean、function、Object、undefined

**6.HTML 中，input 标签的 type 属性有哪些有效的值**

button reset submit

text password checkbox

radio hidden file

**7.在1，2,3，·······，999，1000 中，有多少个数各位数之积等于0**

**8.请列举一些你知道的 SpringMVC 中的注解**

@Controller

@Responsebody

@Service

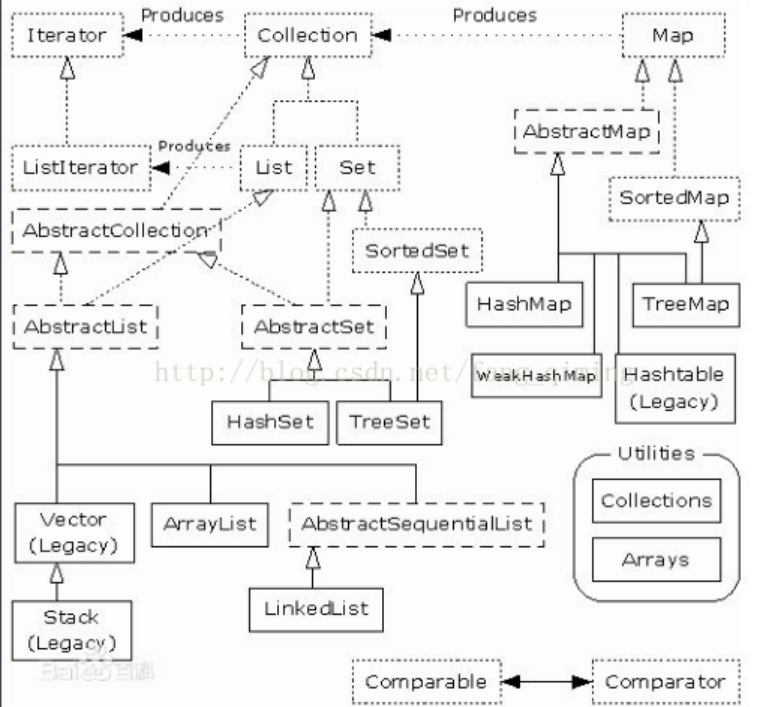
@conpo

@Autowired

@Qualifier

@Resource

**9.请仿照画出 ArrayList 的父子关系图**



**10.下面是一个 java 方法的申明，请猜测该方法的用 途，向面试官确认后实现以下该方法**

public static <T> List<T> except(List<T> list,List<Integer> indexes)

{

Set<Integer> set = new HashSet<>(indexes);

Iterator<Integer> it = set.iterator();

while (it.hasNext()) {

Integer item = it.next();

for (Integer integer : list) {

if (integer.equals(item)) {

it.remove();

}

}

}

for (Integer i : listRepeat) {

System.out.println(i);

}

}

**11.说一说你对 Spring AOP 了解多少吧？**

**12.maven 的打包命令知道吗？**

**13.servlet的生命周期解释一下**

web 容器加载 servlet 并将其实例化后，servlet 生命周期开始，运行其 init() 方法进行 servlet 的初始化，请求到达时运行其 service 方法，service 方法自动派遣运行与请求对应的 doXXX 方法（doGet方法 doPost方法）等，当服务器决定将实例销毁的时候调用其 destroy 方法。

补充：servlet 与 CGI 的区别：

servlet 处于服务器进程中，它通过多线程的方式运行其 service 方法，一个实例可以服务于多个请求，并且其实例一般不会销毁，而 CGI 对每个请求都产生新的进程，服务完成后就销毁，所以效率上低于 servlet。

**14.session 和 application 的区别**

**15.线程池的配置写在哪个文件中？**

HS15春招笔试 https://blog.csdn.net/guodongxiaren/article/details/44539207

**1、 对于 int \*pa[5]; 的描述，正确的是（**D**）**

**A. pa是一个指向数组的指针，所指向的数组是5个int型元素**

**B. pa是一个指向某个数组第5个元素的指针，该元素是int型变量**

**C. pa[5]表示某个数组的第5个元素的值**

**D. pa是一个具有5个元素的指针数组，每个元素是一个int型指针**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：指针数组与数组指针

指针数组：是一个数组，数组中每个元素是一个指针

数组指针：是一个指针，指向一个具有n个元素的数组

这两个概念很容易混淆。其实只要记住优先级就好了：() > [ ] > \*

int \*pa[5]：下标符号[ ] 和优先级大于\*，所以这是一个数组，即指针数组。

int (\*pa)[5]：括号() 的优先级大于下标符号，所以先看括号内的内容，即指针\*，所以这是一个指针，即数组指针。

参考答案：D

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**2、 IPV4中，数据报传播方式包括以下的（C）**

**A. 广播     B. 单播    C. 任播    D. 组播**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：广播、单播、组播

IPV4不支持任播（anycast），IPV6支持。

单播（unicast）：点到点一对一通信。将数据包发送到某一主机

广播（broadcast）：将数据包发送到局域网内所有主机

组播（multicast）：又叫多播，传播范围介于单播和广播之间。将数据包发送到加入同一组的主机

参考答案：C

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**3、 下面关于友元函数的描述中，正确的说法是（**A**）**

**A. 友元函数是独立于当前类的外部函数**

**B. 一个友元函数不可以同时定义为两个类的友元函数**

**C. 友元函数必须在类的外部进行定义**

**D. 在类的外部定义友元函数时必须加上friend关键字**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：友元

友元的出现其实是对面向对象编程思想的一种挑战，因为它破坏OO编程中信息封装的特性，使得一个外部函数可以操纵一个类的私有成员。这样做的好处大概是提高了编程的灵活性并精简了代码。不必像Java那样到处充斥getter和setter方法了。

友元函数并不是类的组成部分，但是它却可以在类内部直接定义，但其作用域并不仅限于类内部，而是和该类的作用域相同，即类外部（但不超出类的作用域的地方）也可见。

友元函数需要访问类的非static成员时，在声明时需要以该类的对象作为参数。

友元参数需要访问类的static成员时，在声明时不需要以该类的对象作为参数。

friend只能用在类内部，在类外定义友元函数内容的时候，不能加friend关键字，否则报错。

友元不是类的组成部分，所以可以定义为多个类的友元（多次重载即可）。

友元样例代码：

#include <iostream>

using namespace std;

class A

{

public:

static int count;

friend void test(A a);

friend void test();

A(int a=1,int b=2):i(a),j(b){};

private:

int i,j;

};

int A::count = 0;

void test(A a)

{

cout<<a.i<<" "<<a.j<<endl;

}

void test(){

cout<<A::count<<endl;

};

int main()

{

A a;

test(a);

test();

}

// 打印结果

1 2

0

参考答案：A

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**4、 以下关于数据库中索引的描述正确的是（C、D）**

**A. 只要允许，任何时候都应该添加索引以加快查询速度**

**B. 无论记录多少，使用索引都能给查询带来性能提升**

**C. 每次update/ insert/ delete操作都会导致索引被重新更新**

**D. 索引可以避免全表扫描**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：索引

数据库的索引并非越多越好，索引的增加会提交查询速度，但是会减慢数据库的修改速度，因为每一次修改都有更新索引，此外还会浪费存储空间（用于维护索引信息）。

此外，我个人认为：记录不多的情况下，使用索引并不能带来性能提升。

参考答案：CD

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**5、 事务的原子性是指（B）**

**A. 事务一旦提交，对数据库的改变是永久的**

**B. 事务中包括的所有操作要么都做，要么都不做**

**C. 一个事务内部的操作及使用的数据对并发的其他事务时隔离的**

**D. 事务必须使数据库从一个一致性状态变到另一个一致性状态**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：ACID、事务特性

原子性（Atomicity）：事务是数据库的逻辑工作单位，事务中包括的诸操作要么都做，要们都不做。

一致性（Consistency）：事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态。

隔离性（Isolation）：对并发执行而言，一个事务的执行不能被其他事务干扰。

持续性（Durability）：一个事务一旦提交，它对数据库中的数据的改变就应该是永久性的。

参考答案：B

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**6、 时间复杂度为O(nlog2n)的排序算法有（**A、B**）**

**A. 快速排序 B. 堆排序**

**C. 冒泡排序 D. 折半插入排序**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：排序算法

在常见排序算法中，时间复杂度为O(nlog2n) 【这里指的是n乘以log以2为底的n】的有：快速排序、归并排序、堆排序

折半插入排序虽然减少了比较顺序，性能较一般的插入排序得到了提升，但是其时间复杂度不变，仍然是O(n^2)。

参考答案：AB

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**7、 软件工程师一种（**B**）分阶段实现的软件工程开发的方法**

**A. 自底向上**

**B. 自顶向下**

**C. 逐步求精**

**D. 面向数据流**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：软件工程

参考答案：B

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**8、 Linux操作系统中有如下的文件结构：**

**$HOME/aaa**

**$HOME/aaa/bb/a.txt**

**$HOME/aaa/my.c**

**请问如果要删除aaa文件夹，要执行的操作是：（**A**）**

**A. rm -rf $HOME/aaa**

**B. rmdir -f $HOME/aaa**

**C. rmdir $HOME/aaa**

**D. rm $HOME/aaa**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：rm、rmdir

很多人可能会误选rmdir。rmdir虽然从字面上理解是删除文件夹，但是它是删除空文件夹。并且rmdir 没有-f这个选项

rm的两个选项：-r 表示递归，即删除文件夹及其子文件和子文件夹。-f 表示force即暴力，无视提示信息。由题干可知，该文件夹并非空文件夹。

参考答案：A

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**9、 执行如下程序代码后，sum的值是（**A**）**

**char chr = 127;**

**int sum = 200;**

**chr += 1;**

**sum += chr;**

**A. 72 B. 99 C. 328 D. 327**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：溢出

char类型可以看作是一个1个字节（8个位）的低精度整型。以整数角度来看，它能表示的整数范围是：-128~127

当chr += 1之后，此时chr的值变成128，但是这时候发生了溢出。此时要得出其实际的值用 128-256 = -128

所以sum+=char是200-128=72

参考答案：A

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**10、 开发一个逻辑模型：公司有10个部门，每个部门有6-7名员工，但每个员工可能会为不止一个部门工作，下面所给的模型正确的是（C）**

**A. 部门和员工之间是一种确定的一对多的关系**

**B. 建立一个关联表，从该关联表到员工建立一个一对多的关系，然后再从该关联表到部门表建立一个一对多的关系**

**C. 建立一个关联表，从员工表到该关联表建立以一对多的关系，然后再从部门表到该关联表建立一个一对多的关系**

**D. 这种情况下不能建立正常的数据库模型**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：多对多关系建表、关联表

解决多对多关系的建表方案，就是建立关联表。本题中，可以建立一个员工表（不包括部门id字段）、部门表（不包括员工id字段）。而关联表至少包括两个字段：员工id和部门id。如果还要给关联表增加一个字段的话就是单独新增一个关联id用做主键。

这样部门增加了员工，或员工加入了新的部门，只需要更新关联表就可了。

参考答案：C

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**11、 下面可用于字符串复制的函数有（A、**E**）**

**A. strcpy B. sprint C. strcmpD. strstrE. memcpy**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：字符串、库函数

**B无此函数（可能是sprintf打印错误），C是字符串比较函数，D是求子串的函数**

**memcpy内存复制;**

参考答案：AE

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**12、 指出下列代码的缺陷（B、C）**

**float f[10];**

**// 假设这里有对f进行初始化的代码**

**...**

**// for循环需要遍历f中所有的元素**

**for (int i = 0; i < 10; )**

**{**

**if (f[++i] == 0)**

**break;**

**}**

**A. for (int i = 0; i < 10; ) 这一行写错了**

**B. f是float型数据直接做相等判断有风险**

**C. f[++i] 应该是 f[i++]**

**D. 没有缺陷**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：浮点型、自增

主要问题：

因为要遍历所有元素，所以不能使用++i

浮点型不能向整型那样直接比较是否相等。

可选方案：

#include <math.h>

...

if (fabs(f[i++]) < 1e-5)

...

// 或者

#define ACCU 1e-5

...

if (f[i++] > -ACCU && f[i++] < ACCU)

...

精度也可以是1e-6。

参考答案：BC

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**13、 以下对C语言的有关描述中，正确的有：（B、C、E）**

**A. 在C程序中，函数调用不能出现在表达式语句中**

**B. 在C语言中，一个函数一般由两个部分组成，它们是函数首部和函数体**

**C. 函数和实参和形参可以是相同的名字**

**D. 在main()中定义的变量都可以在其它被调函数中直接使用**

**E. 外部类型的变量只能定义一次，但可以在不同的地方声明**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：函数、变量

参考答案：BCE

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**14、 有以下函数，该函数的功能是（C）**

**int fun(char \*s)**

**{**

**char \*t = s;**

**while(\*t++);**

**return (t - s);**

**}**

**A. 比较两个字符的大小**

**B. 计算s所指字符串占用内存字节的个数**

**C. 计算s所指字符串的长度**

**D. 将s所指的字符串复制到字符串t中**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：指针与自增

就优先级而言，自增运算符++大于星号\*。但是这个++是在变量名后面的，所以是在该复合表达式的值返回之后，再进行自增。

设p = \*t++; 则其等价于p = \*t; t++。所以while(\*t++)的意思是判断当前字符是否为NULL，如果是就终止循环（是不是NULL，t 都会自增）

如果是p = (\*t)++; 就不一样了。其等价于p = \*t; (\*p)++。在指针所指向的值返回之后，该值加1。

注意当字节是'\0'是停止。最后计算字符偏移的时候，算上了'\0'。

参考答案：C

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**15、 下述程序的执行结果为（A）**

**#include <stdio.h>**

**void abc(char \*str)**

**{**

**int a,b;**

**for(a = b = 0;str[a] != '\0'; a++)**

**if(str[a] != 'c')**

**str[b++] = str[a];**

**str[b] = '\0';**

**}**

**void main()**

**{**

**char str[] = "abcdef";**

**abc(str);**

**printf("str[] = %s",str);**

**}**

**A. str[] = abdef**

**B. str[] = abcdef**

**C. str[] = a**

**D. str[] = ab**

--------------------------------------------------------------------------------------------------------------------------------------------------------------

关键词：字符串、字符指针

参考答案：A

--------------------------------------------------------------------------------------------------------------------------------------------------------------

HS16编程题 https://blog.csdn.net/xiaohui\_loveless/article/details/49691099

**有两有序的数组aArray[] ,bArray[]，已知他们的长度分别为aLength，bLength，求两数组的交集和交集个数**

1. **public** LinkedList<Integer> intersection(**int**[] A, **int**[] B) {
2. **if** (A == **null** || B == **null** || A.length == 0 || B.length == 0) **return** **null**;
3. LinkedList<Integer> list = **new** LinkedList<Integer>();
4. **int** pointerA = 0;
5. **int** pointerB = 0;
6. **while** (pointerA < A.length && pointerB < B.length) {
7. **if** (A[pointerA] < B[pointerB]) pointerA++;
8. **else** **if** (A[pointerA] > B[pointerB]) pointerB++;
9. **else** {
10. list.add(A[pointerA]);
11. pointerA++;
12. pointerB++;
13. }
14. }
16. **return** list;
18. }

复杂度O（N+M）

HS16秋招算法题 https://blog.csdn.net/weixin\_35663229/article/details/52981784

**求出1到100之间所有素数，要求时间复杂度最优。**

快速线性素数筛法（欧拉筛法），时间复杂度O(n)。

诀窍在于：筛除合数时，保证每个合数只会被它的最小质因数筛去。因此每个数只会被标记一次，所以算法时间复杂度为O(n)。

具体请看下面的代码，主要函数是Prime(n)。

#include <bits/stdc++.h>

using namespace std;

vector<int> Prime(int n) { // 求解n以内(含n)的素数

bool flag[n + 1]; // 标记数组,flag[I ]==0表示i为素数,flag[i]==1表示i为合数

memset(flag, 0, sizeof(flag));

vector<int> prime;

int cnt = 0; // 素数个数

for (int i = 2; i <= n; ++i) {

if (!flag[i]) {

prime.push\_back(i); // 将i加入素数表

cnt++;

}

for (int j = 0; j < cnt; ++j) { // 保证每个合数只会被它的最小质因数筛去

if (i \* prime[j] > n) break;

flag[i \* prime[j]] = 1;

if (i % prime[j] == 0) break;

}

}

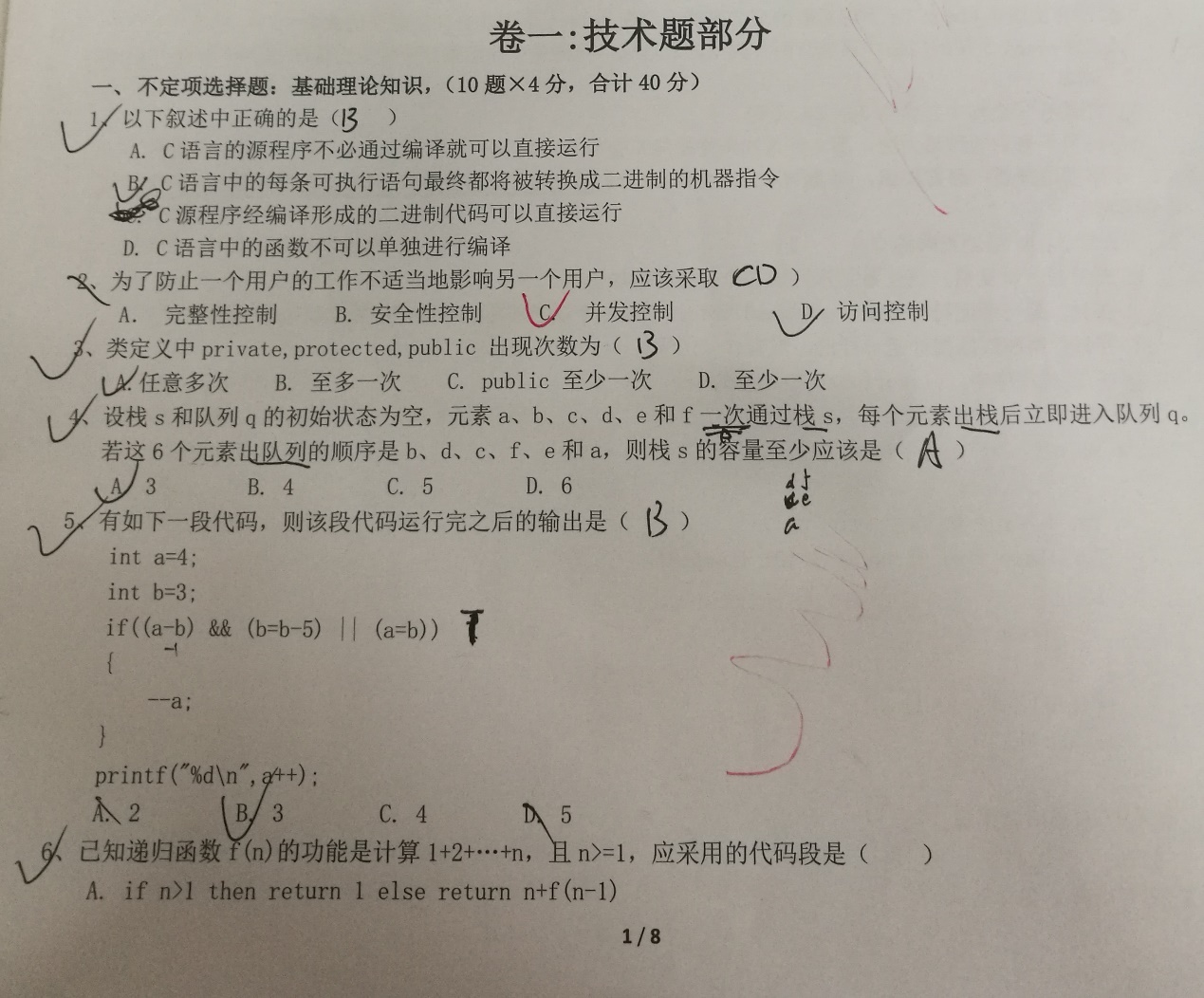
return prime;

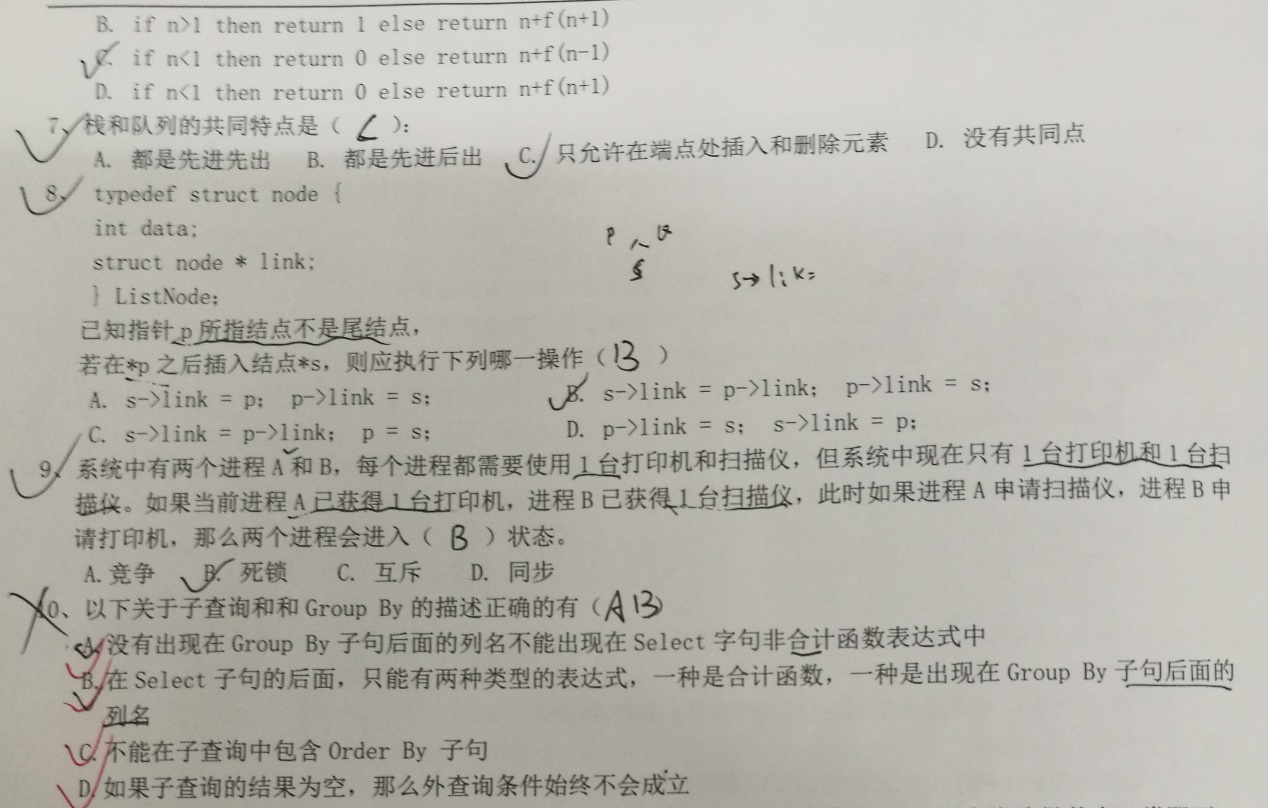
}

**知道1到100天的股票价格，你最多能进行两次买入卖出，求收益最大是多少。（从第二天开始进行买卖操作）**

个人想法：建立数组存放可能的收益；最开始的设为买入点，之后遇到更高的卖出点则更新收益，如果遇到比此买入点更低的，则结束收益并创建新的买入点（贪心）；最后将所有可能的收益排序选前两次。

HS17年笔试 https://blog.csdn.net/nameofcsdn/article/details/72453649



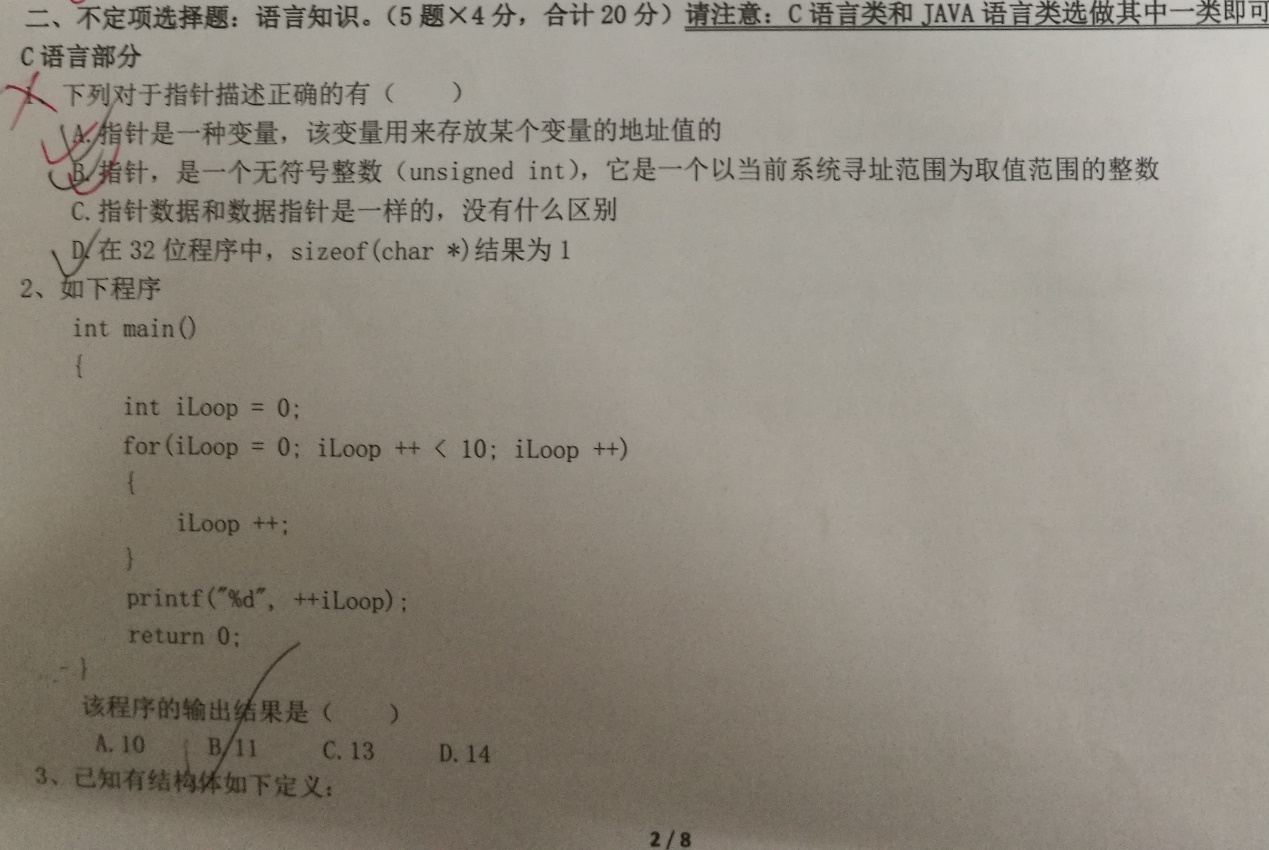


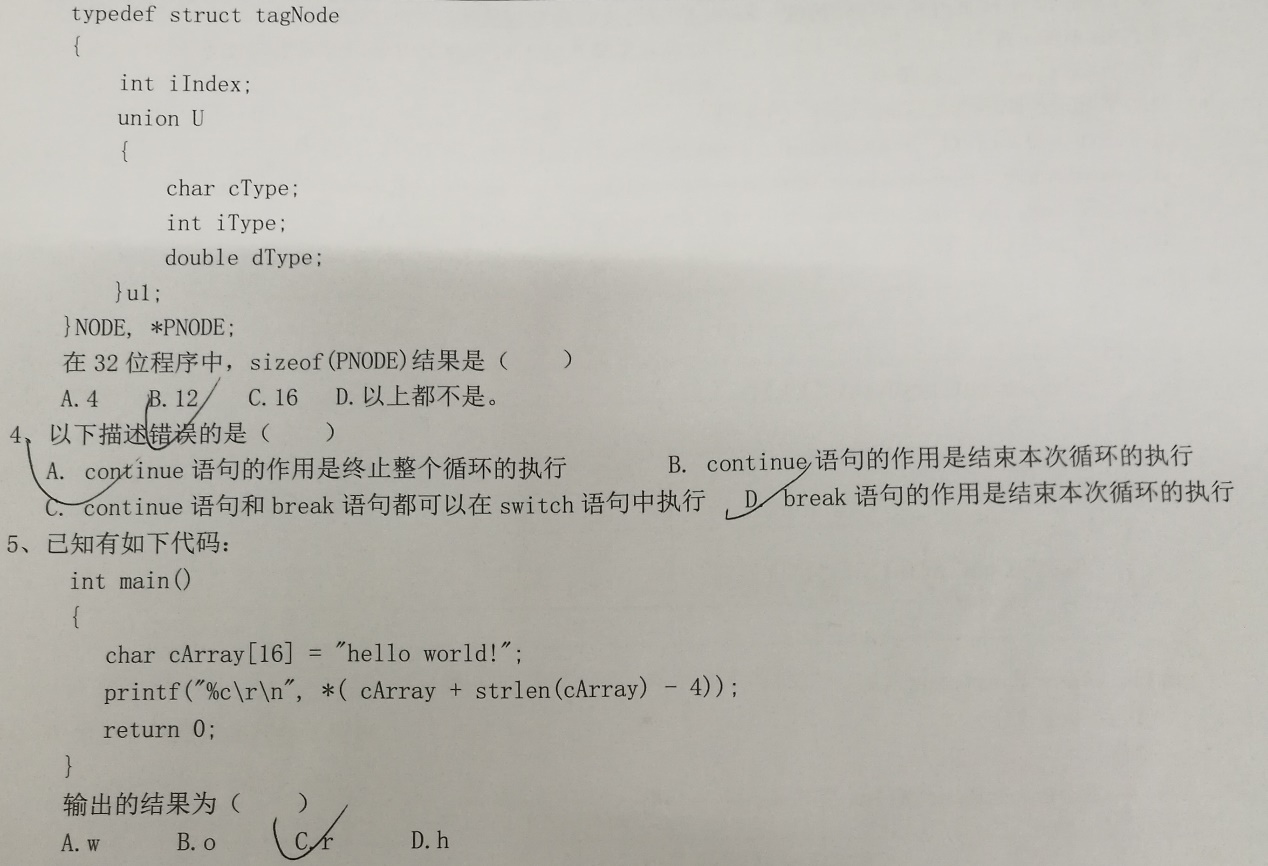
答案：

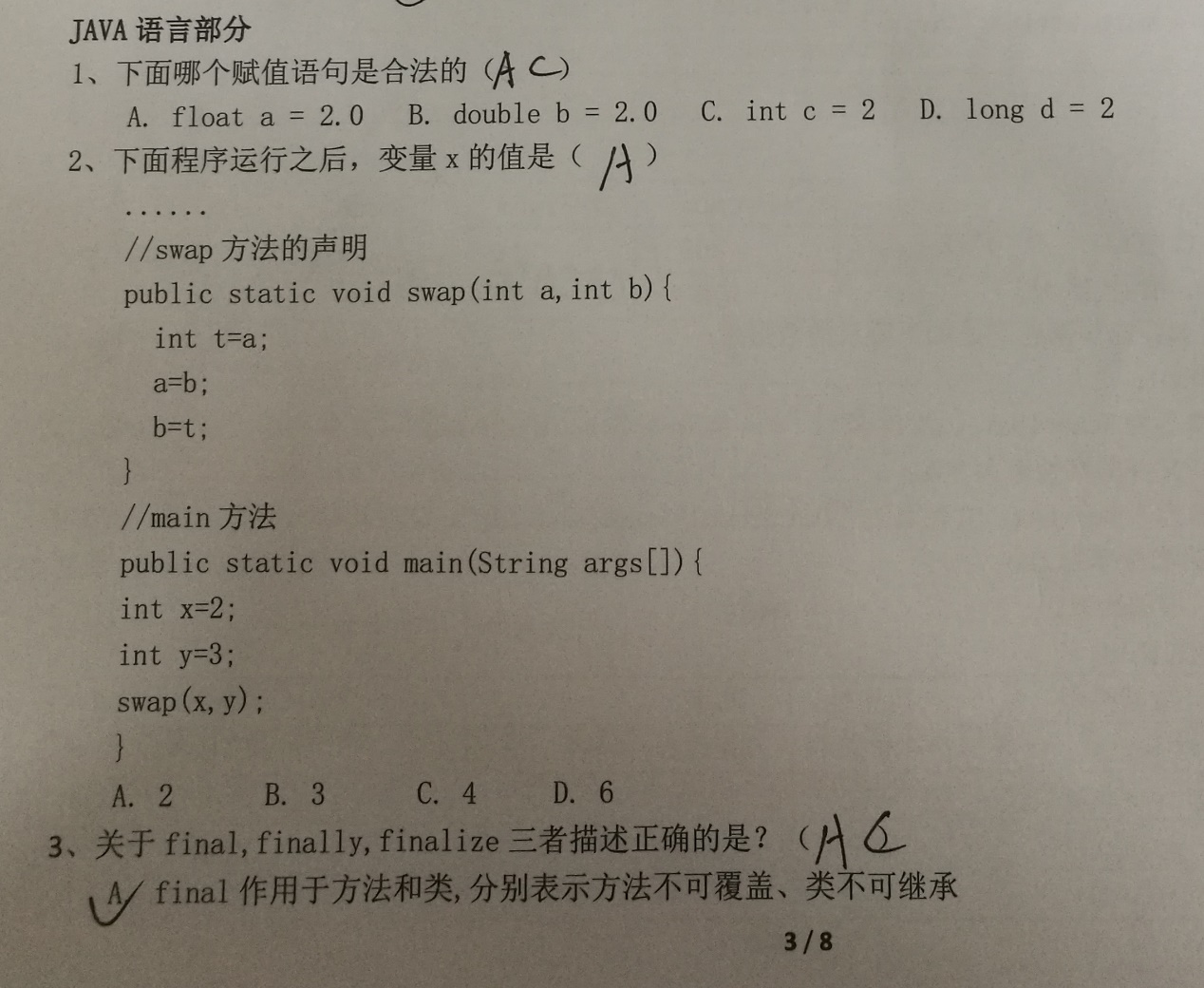
1-5 BCAAB

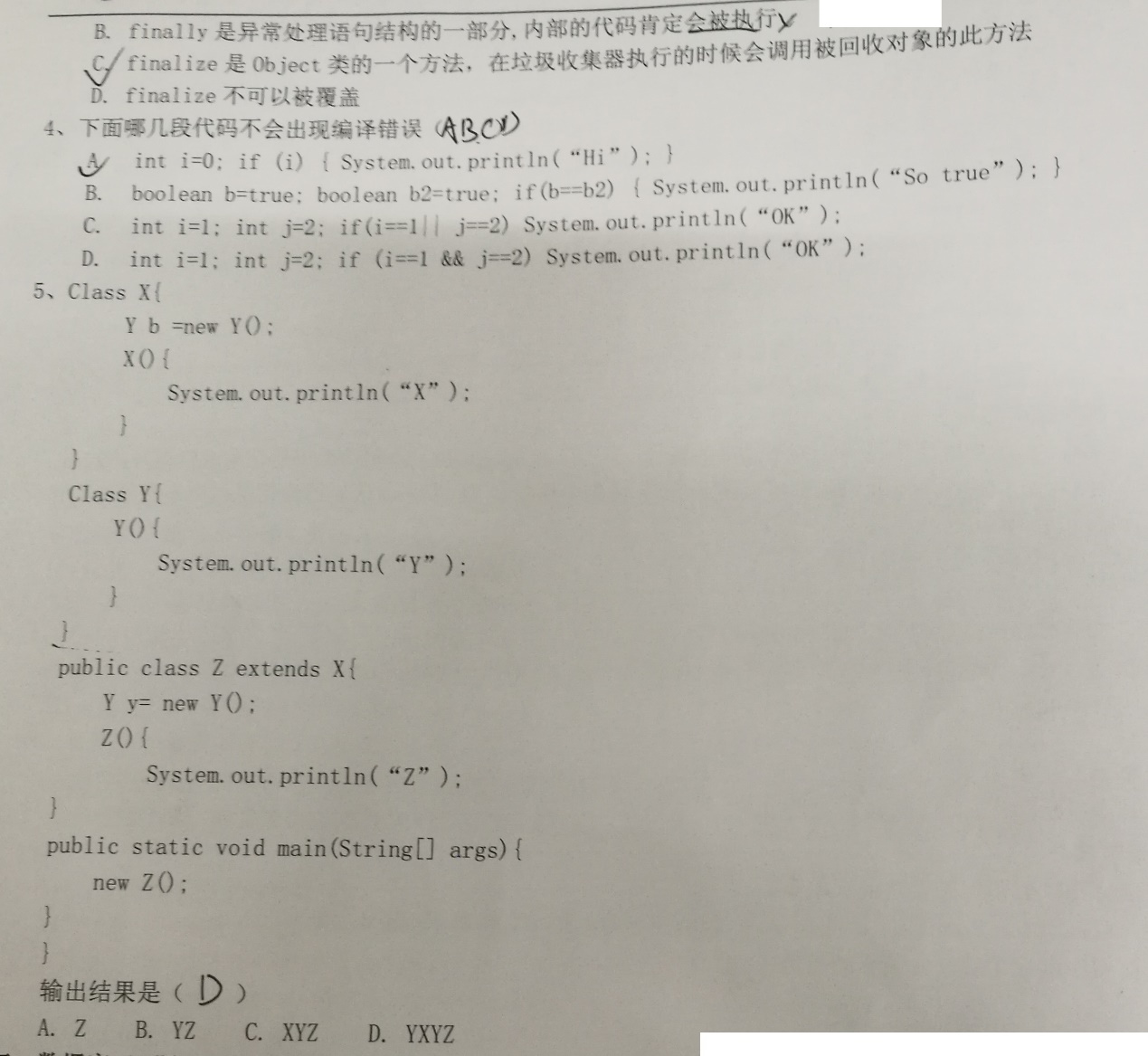
6-9 CCBB

10 ABCD



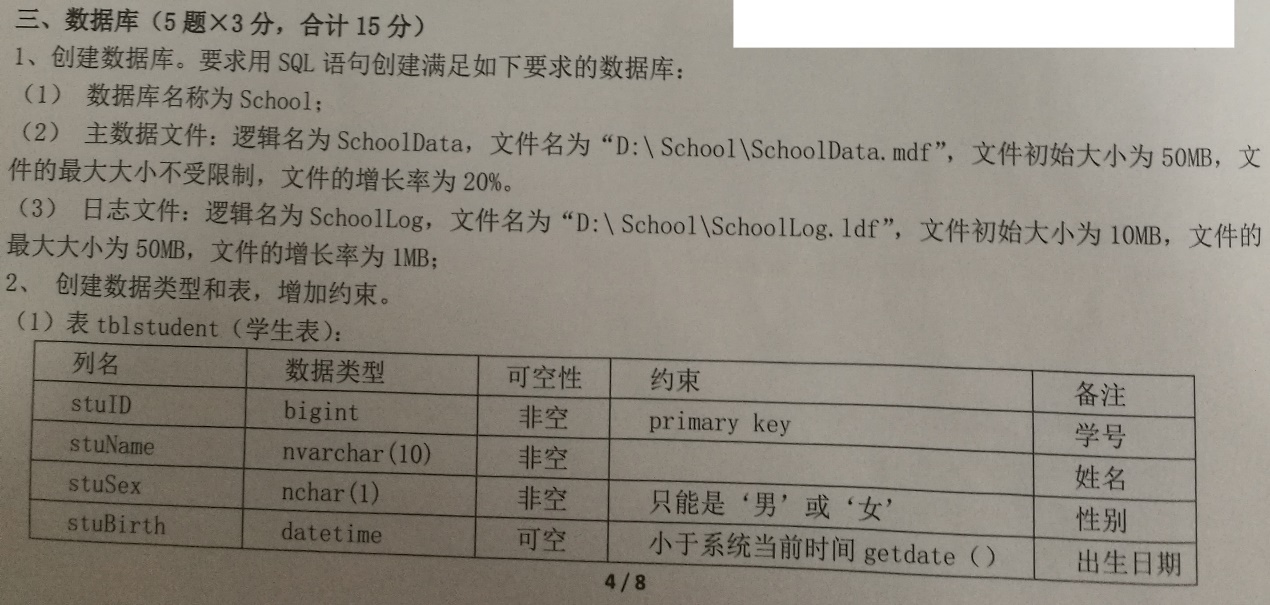






c语言参考：AB D A AD C

Java: BC A AC ABCD D



1.create database School(character set gbk)

on Primary(

name=’SchoolData’,

filename=’D:\School\SchoolData.mdf’,

size=50MB,

(maxsize=),

filegrowth=20%

),

##次数据文件  
filegroup fg(

filename=’..\xxx.ndf’

)

log on(

name=’SchollLog’,

filename=’D:\School\SchoolLog.ldf’,

size=10MB,

maxsize=50MB,

filegrowth=1MB

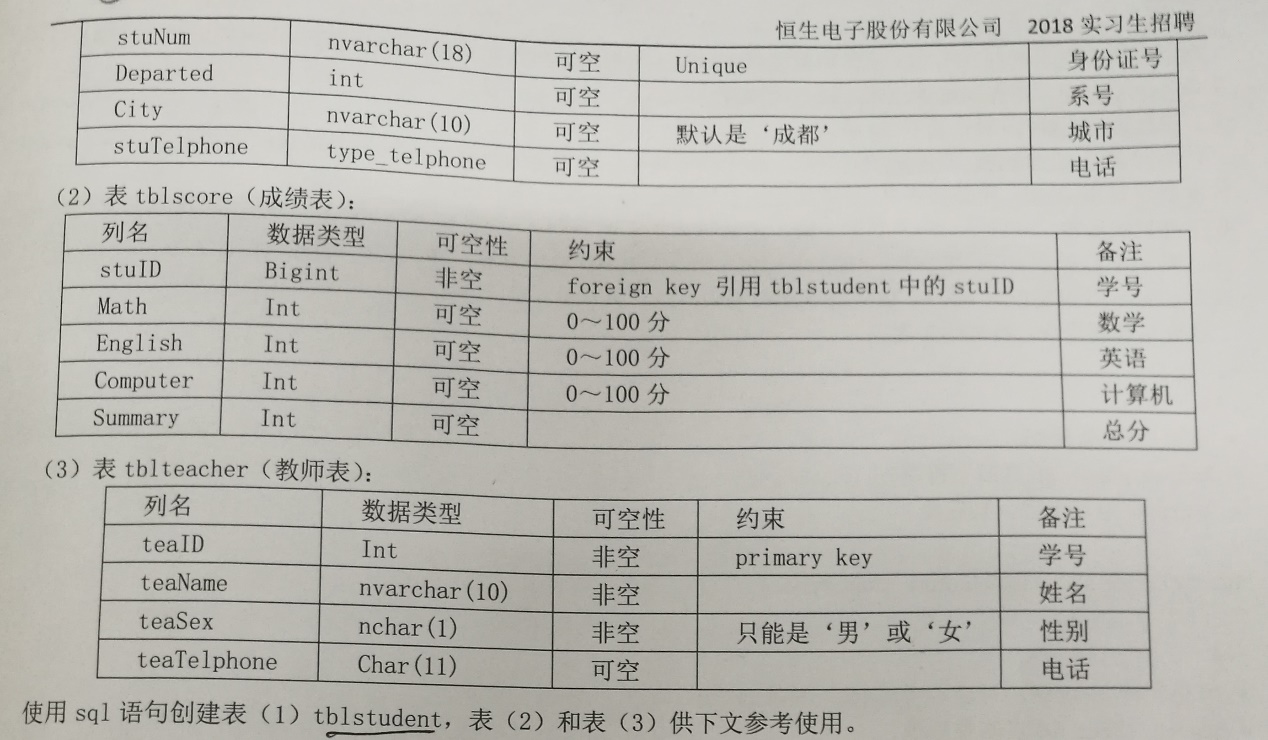
),

##日志2

(

)

go

  
2.create table tblstudent(

stuID bigint primary key comment ’学号’,

stuName nvarchar(10) not null comment ’姓名’,

stuSex nchar(1) not null check(stuSex in(‘男’,’女’) comment ’性别’,

stuBirth datatime check(datatime<getdata()) comment ’出生日期’,

stuNum nvarchar(18) unique comment ’身份证号’,

Departed int comment ’系号’,

City nvarchar(10) default ‘成都’ comment ’城市’,

stuTelphone type\_telphone comment ‘电话’

);

create table tblscore(

stuID bigint not null comment’学号’,

Math int check(Math>=0 and Math<=100) comment ‘数学’,

English int check(English>=0 and English<=100) comment’英语’,

Computer int check(Computer>=0 and Computer<=100) comment’计算机’,

Summary int check(Summary>=0 and Summary<=100) comment’总分’

foretgn key(stuID) references tblstudent(stuID) on update cascade on delete set null

);

create table tblteacher(

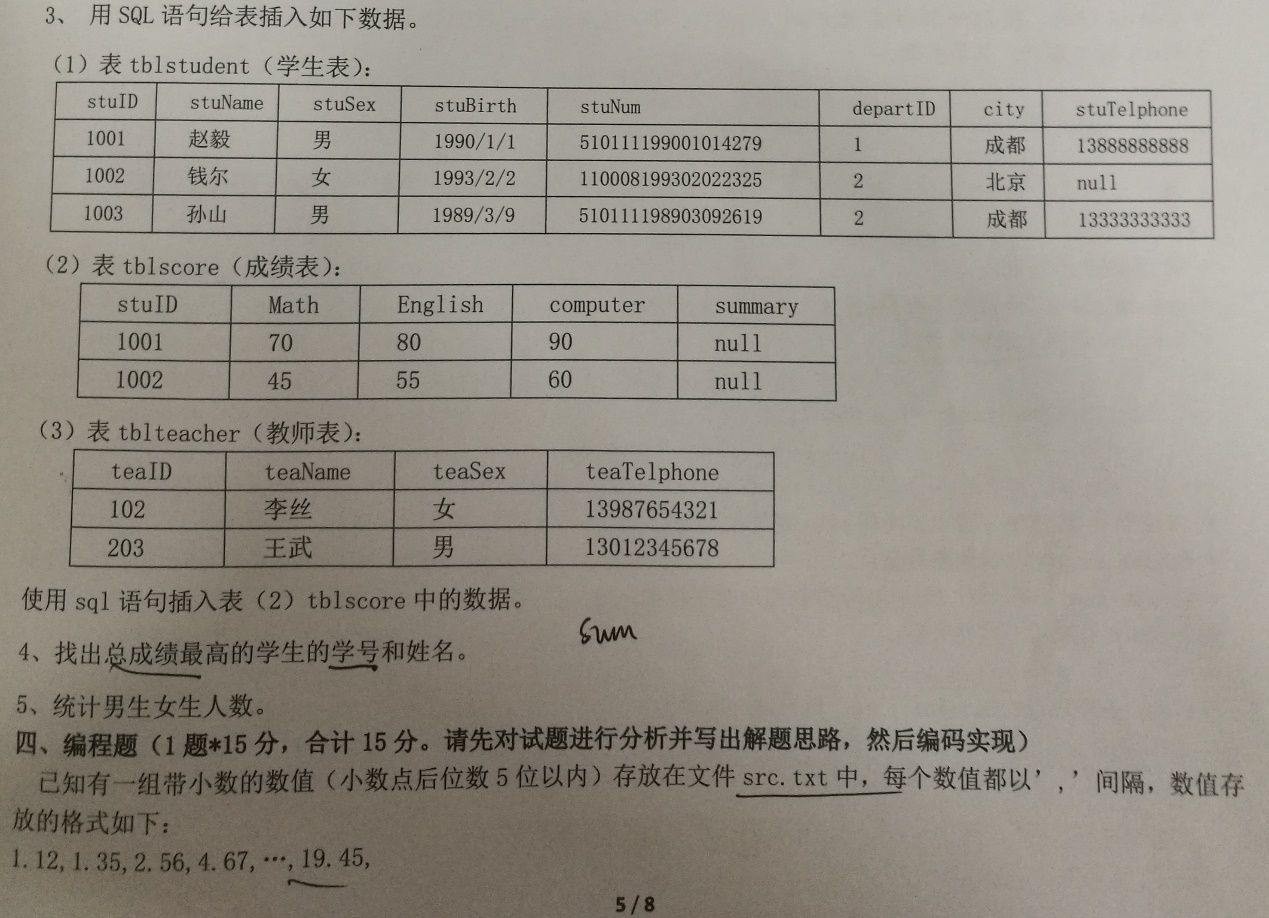
teaID int primary key comment ‘学号’,

teaName nvarchar(10) not null comment ‘姓名’,

teaSex nchar(1) not null check(teaSex in(‘男’,’女’) comment ‘性别’,

teaTelphone char(11) comment ‘电话’

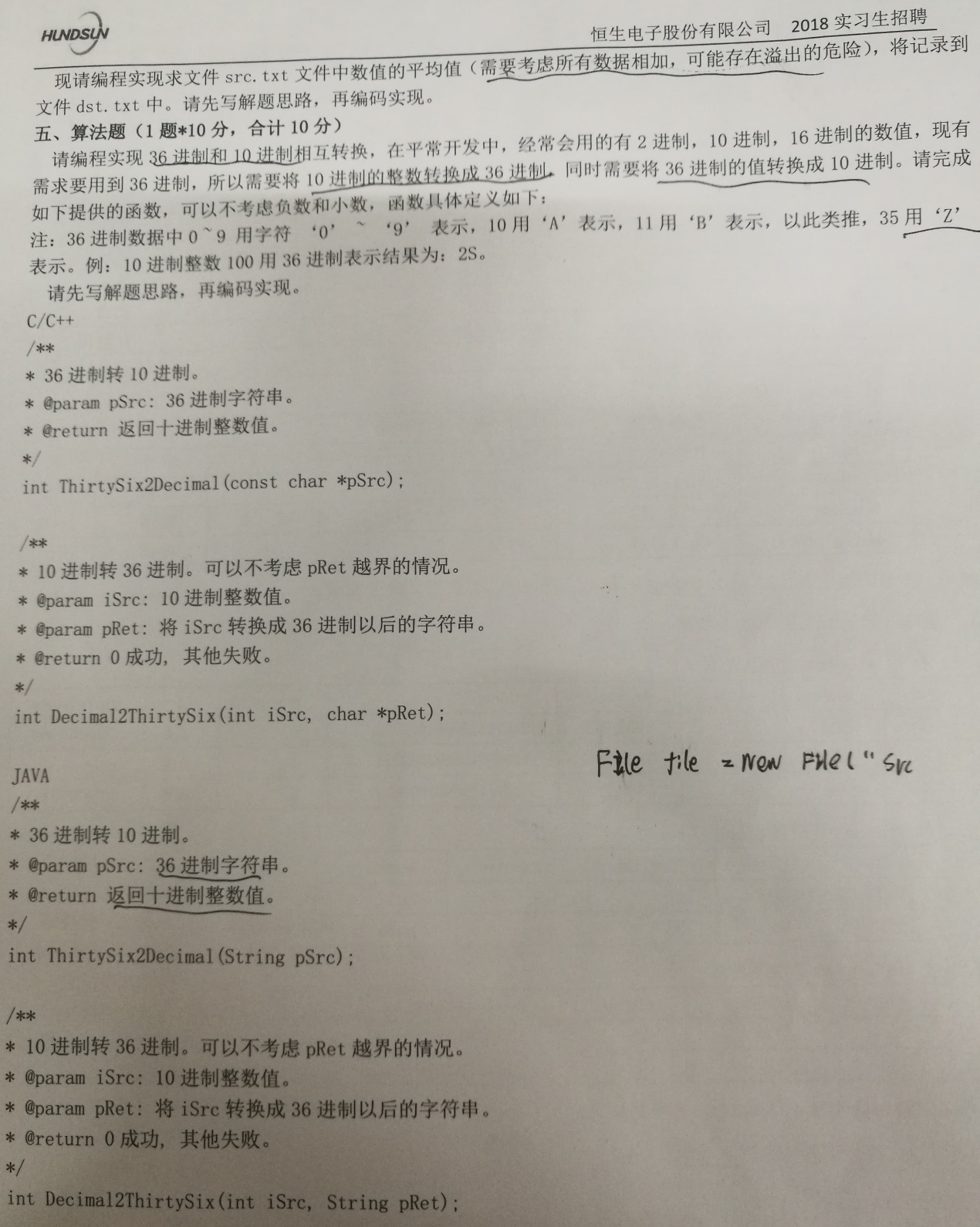
);

  
3.insert into tblscore(stuID,Math,English,computer,summary) values(1001,70,80,90,null);

insert into tblscore values(1002,45,55,60,null);

4.select a.stuID,a.stuName from tblstudent a,tblscore b where a.stuID=b.stuID and b.Summary>=(select Summary from tblscore);

5.select a.stuSex,atotal+btotal as allCount from (select stuSex,count(\*) atotal from tblstudent group by stuSex) a,(select teaSex,count(\*) btotal from tblteacher group by teaSex) b where a.stuSex=b.teaSex;

  
四.

平均值溢出避免：temp = (a&b)+((a^b)>>1)

average = temp + ((temp>>31)&1&(a^b))

逐个求均avg+= (x-avg)/i;

#include<stdio.h>

#include<stdlib.h>

int main()

{

FILE \*fp;

int i=0,j=0,count;

float array[100], avg=0;

char ch=’’ ;

fp=fopen("src.txt","r");

if(fp==NULL)

{

printf("cannot open the file!");

exit(0);

}

//从src.txt中读取浮点型数据，数据之间用逗号隔开

while(!feof(fp))

{

fscanf(fp,"%f%c",&array[i++],&ch);//ch是用来保存逗号字符

}

count=i;

fclose(fp);

avg=array[0];

for(j=1;j<count;j++)

{

avg+=(avg-array[j])/count;

}

//将平均值保存到dst.txt文件中

fp=fopen("dst.txt","w");

if(fp==NULL)

{

printf("cannot open the file!");

exit(0);

}

fprintf(fp,"%f",avg);//将avg输入到fp指向的磁盘文件"dst.txt"中

fclose(fp);//打开文件，一定要记得关闭文件！

return 0;

}

五.

import java.util.HashMap;

/\*\*

\* 36进制与10进制转换思路：

\* 创建HashMap类型对象用于存放数字'0'到字母'Z'36个字符值键对

\*/

public class Ten2Thirty {

//定义36进制数字

private static final String X36 = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";

//拿到36进制转换10进制的值键对

private static HashMap<Character, Integer> thirysixToTen = createMapThirtysixToTen();

//拿到10进制转换36进制的值键对

private static HashMap<Integer,Character> tenToThirtysix = createMapTenToThirtysix();

//定义静态进制数

private static int BASE = 36;

//用来存放10转36进制的字符串

private static StringBuffer sb = new StringBuffer();

private static HashMap<Character, Integer> createMapThirtysixToTen() {

HashMap<Character, Integer> map = new HashMap<Character, Integer>();

for (int i = 0; i < X36.length(); i++) {

//0--0,... ..., Z -- 35的对应存放进去

map.put(X36.charAt(i), i);//String.valueOf(i)

}

return map;

}

private static HashMap<Integer, Character> createMapTenToThirtysix() {

HashMap<Integer, Character> map = new HashMap<Integer,Character>();

for (int i = 0; i < X36.length(); i ++) {

//0--0,... ..., 35 -- Z的对应存放进去

map.put(i, X36.charAt(i));

}

return map;

}

/\*\* 36 to 10

\* @param pStr 36进制字符串

\* @return 十进制

\*/

public static int ThirtysixToDeciaml(String pStr) {

if (pStr == "") return 0;

//目标十进制数初始化为0

int deciaml = 0;

//记录次方,初始为36进制长度 -1

int power = pStr.length() - 1;

//将36进制字符串转换成char[]

char[] keys = pStr.toCharArray();

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

//拿到36进制对应的10进制数

int value = thirysixToTen.get(keys[i]);

deciaml = (int) (deciaml + value\*Math.pow(BASE, power));

//执行完毕 次方自减

power --;

}

return deciaml;

}

/\*\*

\* 用递归来实现10 to 36

\* @param iSrc

\* @return

\*/

public static String DeciamlToThirtySix(int iSrc) {

//循环控制

boolean flag = true;

//定义键

int key;

//定义值

int value;

while (flag) {

key = iSrc / *BASE*;

value = iSrc % *BASE*;

if (key != 0) {

//递归

*DeciamlToThirtySix*(key);

}

//输出各位数

*sb*.append(*tenToThirtysix*.get(value).toString());

flag = false;

}

return sb.toString();

}

public static void main(String[] args) {

int x = ThirtysixToDeciaml("2S");

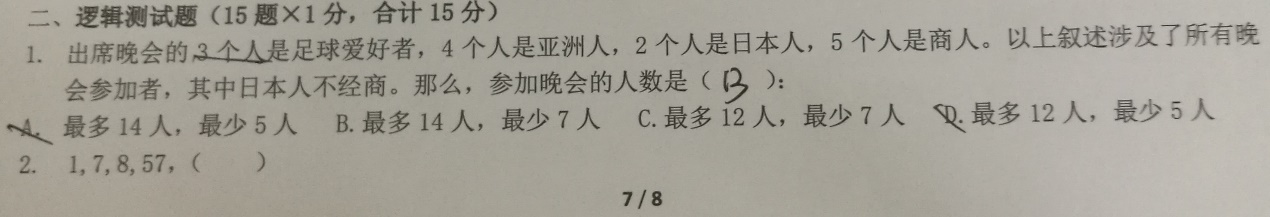
String s = DeciamlToThirtySix(100);

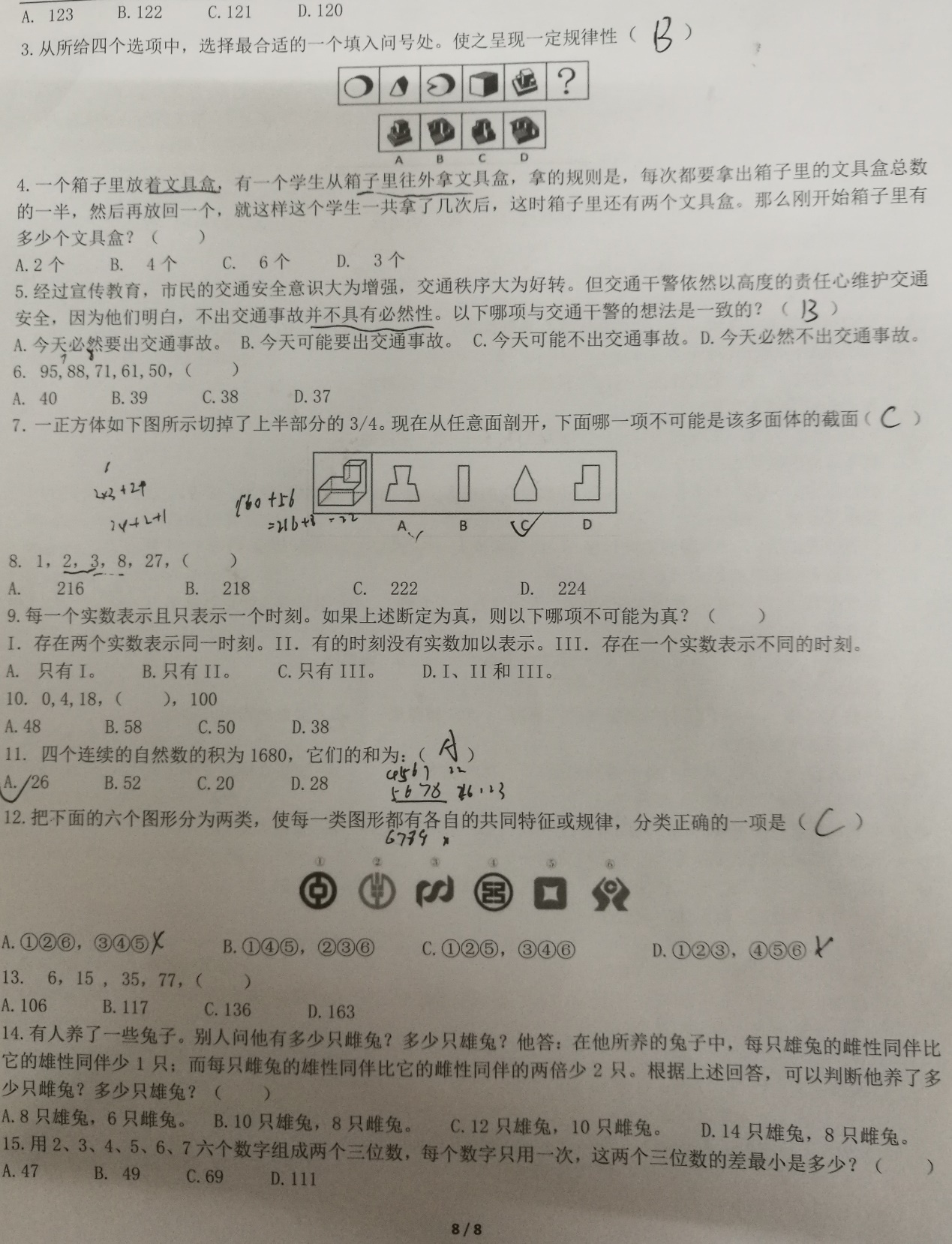
System.out.println(s);

}

}

下面是智力测试题





参考：

B、C、B、A、B、C、C、D、C、A、B、D、A、A

[恒生电子的一道编程题](https://lelglin.iteye.com/blog/1222852) https://lelglin.iteye.com/blog/1222852

题目：4\*8的一块格子，设x为0-8，y为0-4；按照象棋中‘马’的步伐走，即能从（0,0）走到（1,2）或者（2,1）。

假设只能朝x增大的方向走，不能回头，一直走到x=8，输出所有可能的路线

public class ce{

static int *index*=0;、】

static String *location*[]=new String[9];

public static void jumpTo(int x,int y){

if(x==8){

for(int i=0;i<=*index*;i++){System.*out*.print(*location*[i]);}

System.*out*.println();

}else{

if(y+2<=4){*index*++; *location*[*index*]="("+(x+1)+","+(y+2)+")"; *jumpTo*(x+1,y+2); *index*--;}

if(y-2>=0){*index*++; *location*[*index*]="("+(x+1)+","+(y-2)+")"; *jumpTo*(x+1,y-2); *index*--;}

if(x+2<=8){

if(y+1<=4){*index*++; *location*[*index*]="("+(x+2)+","+(y+1)+")"; *jumpTo*(x+2,y+1); *index*--;}

if(y-1>=0){*index*++; *location*[*index*]="("+(x+2)+","+(y-1)+")"; *jumpTo*(x+2,y-1); *index*--;}

}

}

}

public static void main(String[] args){

*location*[0]="(0,0)";

*jumpTo*(0,0);

}

}

恒生电子笔试题目 <https://wenku.baidu.com/view/8c8eed3e0622192e453610661ed9ad51f01d54e3.html>

以下纯属参考，未必这么考honglei：请大家一定要注意oracle数据库方面的知识现在用人单位在java和c++的笔试上基本上难不倒我们但在一些对于数据库有比较高要求的软件企业，比如做金融很多时候，我们会有oracle数据库方面受挫，从而造成在薪资水平上受压制，甚至失去这个机会。

1.对于一个数据库表，可以有1个主键和多个外键。

2．为了防止一个用户的工作不适当的影响另一个用户，应该采取（c）a,完整控制b，安全性控制c，并发控制d，访问控制

3．关系数据库中，实现实体之间的联系是通过表与表之间的（d）a，公共索引b，公共存储c，公共元组d，公共属性

4，sql语言集数据查询，数据操纵，数据定义，数据控制功能与一体，充分体现了关系数据库语言的特点和优点。

5，触发器可以在插入，删除或修改特定表中的数据中的一种或几种操作发生时自动执行。6，请列数据库编程中连接sqlserver数据库的三中不同技术：dsn，dsn-less，oledb

7，请举出实体间具有一对一，一对多，多对多联系的例子。

一对一:班级与班长之间的联系：  
一个班级只有一个正班长  
一个班长只在一个班中任职

一对多：班级与学生之间的联系：  
一个班级中有若干名学生，  
每个学生只在一个班级中学习

多对多：课程与学生之间的联系：  
一门课程同时有若干个学生选修  
一个学生可以同时选修多门课程

8，简要说明事务的原子性？

事务的原子性指的是，事务中包含的程序作为数据库的逻辑工作单位，它所做的对数据修改操作要么全部执行，要么完全不执行。这种特性称为原子性。

事务的原子性要求，如果把一个事务可看作是一个程序，它要么完整的被执行，要么完全不执行。就是说事务的操纵序列或者完全应用到数据库或者完全不影响数据库。这种特性称为原子性。

假如用户在一个事务内完成了对数据库的更新，这时所有的更新对外部世界必须是可见的，或者完全没有更新。前者称事务已提交，后者称事务撤消（或流产）。dbms必须确保由成功提交的事务完成的所有操纵在数据库内有完全的反映，而失败的事务对数据库完全没有影响。

9，简要说明为什么存储过程执行速度比普通的sql更快而且减少网络流量？

谁都知道存储过程是预编译的

存储过程其实就是能完成一定操作的一组sql语句，只不过这组语句是放在数据库中的(这里我们只谈sqlserver)。如果我们通过创建存储过程以及在asp中调用存储过程，就可以避免将sql语句同asp代码混杂在一起。这样做的好处至少有三个：d$y/

第一、大大提高效率。存储过程本身的执行速度非常快，而且，调用存储过程可以大大减少同数据库的交互次数。

第二、提高安全性。假如将sql语句混合在asp代码中，一旦代码失密，同时也就意味着库结构失密。、

第三、有利于sql语句的重用。

应用题：0s

student

学号 姓名 性别 年龄 所在系  
sno sname ssex sage sdept

course

课程编号 课程名 学分

cno cname ccredit

sc

学号 课程编号 成绩

sno cno grade

1. 查询所有学过课程“数据库”的学生姓名，并且按年龄从大到小排序

select stu.sname stu.sage from studentstu,course,sc where stu.sno=sc.sno and sc.cno=course.cno and course.cname=‘数据库‘ order by age desc

排名函数 rank()over()

年龄降序排序,若排序字段相同,则排名相等

select \*,rank() over(order by age desc) as 排名 from student

1. 查询平均分数为85以上的学生名单和平均得分

select stu.sname,avgrade from student stu,(select sno,avg(grade) avgrade from sc group by sno having avg(grade)>85)nsc where stu.sno=nsc.sno;

1. 新增加一门2个学分的课程，编原理课程编号为1005（char型）

insert into course(cno,cname,ccredit) values(‘1005’,”编译原理”，2);

1. 在选修关系sc表中，建立序号和课程编号的唯一索引。

create unique index sc\_sno\_cno on sc(sno,cno);

1. 学生信息中，要增加身份证号码字段类型为char长度18非空

alter table student add credit\_id char(18) not null；

1. 删除sc表中学号已经不存在的记录

delete from sc where sno not in(select sno from sc);

1. 写出sql语句得到下列结果

所在系 男生人数 女生人数 总人数

select sdept,count(\*) atotal,sum(case ssex when ‘男’ then 1 else 0 end) as mtotal,sum(case ssex when ‘女’ then 1 else 0 end) as femtotal from student group by sdept;

新建一个系部表department（系部名称，总人数，男生人数，女生人数）,用T-sql语句从学生表中采集相应信息添加到系部表中

介绍两种方法

1.用到了select into，要注意的是select into要求此前并没有department表

完成这题的方法可以为

    select 所在系 as 系部名称,count(\*) as 总人数,

    sum(case 性别 when '男' then 1 else 0 end) as 男生人数,

    sum(case 性别 when '女' then 1 else 0 end) as 女生人数

    into department

    from 学生

    group by 所在系

所在系 15<年龄〈18 18〈年龄〈20的人数 其他年龄的人数 总人数

select sdept,sum(case sage when sage>15 and sage<18 then 1 else 0 end) as bt\_15\_18,sum(case sage when sage>18 and sage<20 then 1 else 0 end) as bt\_18\_20,sum(case sage when sage<=15 and sage>=20 and sage=18 then 1 else 0 end) as others,count(\*) total from sc group by sdept;

第一题是综合题，

1、毕业后想从事什么样的工作，为什么？

2、你父母对你的教育上，哪些造就了你现在的个性特点？

3、你的技术专长是什么？

4、你的学习成绩怎样？拿过什么奖励金？

第二题是不定项选择题

1、给你一串出栈序列，判断栈至少长多少

2、给出先序中序判断后续

3、字符数组赋值问题，包括for循环和memset和另一个dzero（好像是这个函数）4、linux/Unix中允许读写但不允许执行的命令（666跟777）

5、关于静态变量说法正确的是哪些

static表示“全局”或者“静态”的意思，用来修饰成员变量和成员方法，也可以形成静态static代码块，但是Java语言中没有全局变量的概念。

被static修饰的成员变量和成员方法独立于该类的任何对象。也就是说，它不依赖类特定的实例，被类的所有实例共享。

只要这个类被加载，Java虚拟机就能根据类名在运行时数据区的方法区内定找到他们。因此，static对象可以在它的任何对象创建之前访问，无需引用任何对象。

用public修饰的static成员变量和成员方法本质是全局变量和全局方法，当声明它类的对象市，不生成static变量的副本，而是类的所有实例共享同一个static变量。

static变量前可以有private修饰，表示这个变量可以在类的静态代码块中，或者类的其他静态成员方法中使用（当然也可以在非静态成员方法中使用–废话），但是不能在其他类中通过类名来直接引用，这一点很重要。实际上你需要搞明白，private是访问权限限定，static表示不要实例化就可以使用，这样就容易理解多了。static前面加上其它访问权限关键字的效果也以此类推。

6、关于数据库中索引作用的题  
设计数据库中的索引可以大大提高系统的性能：

1、通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。

2、可以大大加快数据的检索速度，这也是创建索引的最主要的原因。

3、可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义。

4、在使用分组和排序 子句进行[数据检索](https://www.baidu.com/s?wd=%E6%95%B0%E6%8D%AE%E6%A3%80%E7%B4%A2&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)时，同样可以显著减少查询中分组和排序的时间。

5、通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

7、进程的并发会影响数据一致性的（读脏数据等）

8、待补

第三题填空题

1. 操作系统进程高级通信有哪些方式

共享存储器、管道文件和消息传递

2、给出一段代码，问这段代码有什么问题（指针为赋初值）

3、给出IP，求子网掩码和可分配的主机数

A类 (空1)2^7-2(网络号)+2^24-2(主机号)

B类 (2)2^14-1 + 2^16-2

C类 (3)2^21-1 + 2^8-2

IP地址：190.15.10.0 子网掩码255.255.240.0（/20），能划分多少个子网，每个子网能容纳多少台主机？

1、255.255.240.0 和/20是一个意思，都是表示掩码有20位为1，12位0。用二进制表示：  
1111 1111.1111 1111.1111 0000.0000 0000  
2、要明白掩码为1的位表示IP地址的位为网络位，为0的位为主机位，所以可以确定网络的IP范围：190.15.0.0--190.15.239.255  
3、如果你的意思是把B类地址划分成了多少个子网，那么计算是根据借了4个主机位作为网络位，4位最多是16个子网。  
4、每个子网，其实就是一个其实就是主机位12位的子网，IP数则为4096个，去掉两个不可分配IP，实际可用IP是4094个。  
4、数据库中事务的四个特点（原子性、一致性、独立性/隔离性、持久性）

上有，略

5、待补

第四题数据库操作题

大致就是给出三个表，要求写SQL语句，包括视图建立，增删改查，SQL函数应用等（一共25分，好多）

create view viewname as select \* from table where 条件 go;

第五题用C或JAVA写一个五子棋程序

要求写一个函数，当一颗白棋落下的时候，判断白方是否五连珠，棋盘是13\*13大小，用数组A表示，0表示空，1表示白棋，2表示黑棋

只列出水平方向的判断

int check\_win(int m, int n)

{

    int total = 1;

    int i;

    for(i = m-1; i >= 0; i --)//统计同行左侧连续同样棋子个数。

        if(map[i][n] == map[m][n]) total++;

        else break;

    for(i = m+1; i < 100; i ++)//统计同行右侧连续同样棋子个数。

        if(map[i][n] == map[m][n]) total++;

        else break;

    if(total>=5) return 1;//胜利。

    return 0; //没有胜利。

}

然后就是一道20分值的数据库的题目了，一个表里有三个字段，语言，数学，英语，三门课程的成绩分别为70、80、58，要你写语句输出三门课程的成绩（不过成绩是要用优秀、及格、不及格来显示的）大于80分为优秀、60分以上为及格、60分以下为不及格。要求你写出设计的思路，然后再写语句。

建表

drop table if exists classestest;

create table classestest (

eid bigint(10) not null auto\_increment,

yuwen int(3) default null,

shuxue int(3) default null,

yingyu int(3) default null,

primary key (eid)

) engine=InnoDB auto\_increment=1 default charset=utf8 comment='学科测试表';

insert into classestest (yuwen,shuxue,yingyu) values (70,80,58);

select (case when yuwen>=80 then '优秀' when yuwen>=60 then '及格' else '不及格' end) as 语文,(case when shuxue>=80 then '优秀' when shuxue>=60 then '及格' else '不及格' end) as 数学,(case when yingyu>=80 then '优秀' when yingyu>=60 then '及格' else '不及格' end) as 英语 from classestest;

最后一部分也是20分值的题，有两道题，任选一道，第一道是看程序填空的，考的是递归算法；第二道是用JAVA写的，实现折半查找。

public int binarySearch(int[] arr, int findElem) {

int low = 0;

int high = arr.length - 1;

int mid;

while (low <= high) {

mid = (low + high) / 2;

//如果要查找的元素findElem小于中间位置的元素mid，指向数组的较大端的high索引重新指向中间索引mid的左边（mid-1）

if (findElem < arr[mid]) {

high = mid - 1;

}

//如果要查找的元素findElem大于中间位置的元素mid，指向数组的较小端的low索引重新指向中间索引mid的右边（mid+1）

if (findElem > arr[mid]) {

low = mid + 1;

}

if (arr[mid] == findElem) {

return mid;

}

}

return -1;

}

}

数据库练习参考<https://wenku.baidu.com/view/ee28118681c758f5f71f676c.html>