1. 选择题（20\*2）

1）对于java的基本类型以下说法正确的是（C）

A. long的占用字节比double大

B. char与byte都占用1字节

C. int与float都占用4字节

D. short范围为-2^7到2^7-1

2）下列属于Java合法的一组标识符是（B）

A. void define "ABC" \_sys\_tea

B. If \_123 $points myName

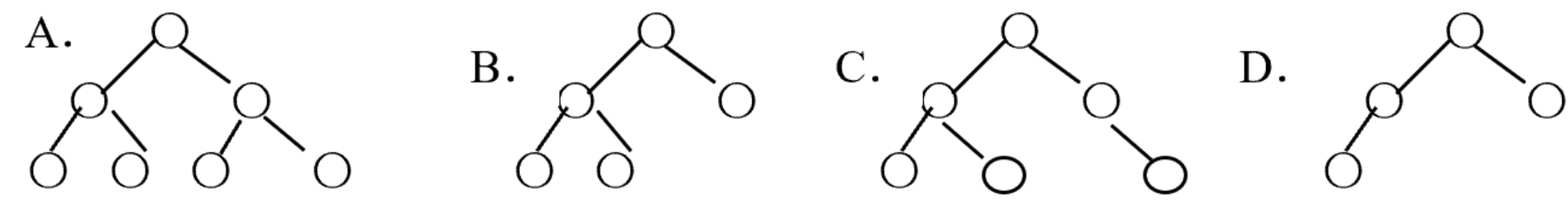
C. &5678 class case #time

D. DO +static 25c My\_name

3）从一个具有n个结点的单链表中查找等于x结点时．在查找成功的情况下，需平均比较( D )个结点.

A. n B. n/2 C. (n-1)/2 D. (n+1)/2

4）如下图所示的4棵二叉树中，( C )不是完全二叉树.



5）设有一个二维数组A\[m][n]，假设\*A\*\[0][0]存放位置在644(10)，\*A\*\[2][2]存放位置在676(10)，每个元素占一个空间，问\*A\*\[3]\[3](10)存放在什么位置？脚注(10)表示用10进制表示。（C）

A．688 B．678 C．692 D．696

6）Spring中定义了一系列的注解，以下有关其常用的注解说明错误的是。（A）

A. @Autowired用于对Bean的属性变量、属性的setter方法及构造方法进行标注，配合对应的注解处理器完成Bean的自动配置工作，默认按照Bean的名称进行装配。

B. @Repository用于将数据访问层（DAO层）的类标识为Spring中的Bean。

C. @Service通常作用在业务层（Service层），用于将业务层的类标识为Spring中的Bean。

D. @Controller通常作用在控制层（如Spring MVC的Controller），用于将控制层的类标识为Spring中的Bean。

7）针对GET方式RESTful风格的请求http://localhost:8080/chapter14/user/1，后台方法接收参数的路径映射写法不可行的是。（B）

A. @RequestMapping(“/user/{id}”,method=RequestMethod.GET)。

B. @RequestMapping(value="/user/ids")。

C. @GetMapping(value="/user/{ids}")。

D. @GetMapping(value="/user/{id}"，method=RequestMethod.GET)

8）System.out.println("5" + 9);运行结果是（A）

A. 59 B.14 C. 2 D. 5

9）System.out.println(5 << 2);运行结果是（D）

A. 3 B.4 C. 16 D. 20

10）System.out.println(-7 & 4);运行结果是（C）

A. 13 B. 5 C. 0 D. -3

11）以下代码执行结果正确的是（D）

int i = 1;

i = i++;

int j = i++;

int k = i + ++i \* i++;

System.out.println(i);

System.out.println(j);

System.out.println(k);

A. 3,2,10 B. 3,3,14 C. 4,1,11 D. 3,3,18

12）以下代码执行结果正确的是（B）

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

list.add(2);

list.add(11);

list.add(14);

List<Integer> collect = list.stream().filter(e -> e > 10).collect(Collectors.toList());

System.out.println(collect);

A. [2] B. [11,14] C. [14] D. [2,11,14]

13）以下代码执行结果正确的是（D）

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

list.add("2");

list.add("11");

list.add("14");

List<String> collect =

list.stream().filter(e -> e.compareTo("10") > 0).collect(Collectors.toList());

System.out.println(collect);

A. [2] B. [11,14] C. [14] D. [2,11,14]

14）执行完以下代码int [ ] x = new int[25]；后，以下哪项说明是正确的（ A ）：

A. x[24]为0

B. x[24]未定义

C. x[25]为0

D. x[0]为空

15）以下代码输出结果为（C）

public static void main(String[] args) {

String name = "张三";

User user = new User(name);

changeName(user, name);

System.out.println(user.getName());

System.out.println(name);

}

static void changeName(User user, String name) {

user.setName("李四");

name = "李四";

}

class User {

private String name;

User(String name) {

this.name = name;

}

public void setName(String name) {

this.name = name;

}

public String getName() {

return name;

}

}

A. 李四，李四 B. 张三，李四 C. 李四，张三 D. 张三，张三

16）以下代码输出正确的是（D）

public static void main(String[] args) {

Cat cat = new Cat();

cat.printType();

Animal animal = (Animal) cat;

animal.printType();

}

public static class Animal {

public void printType() {

System.out.println("Animal");

}

}

public static class Cat extends Animal {

public void printType() {

System.out.println("Cat");

}

}

A. Cat，Animal B. Animal，Cat C. Animal，Animal D. Cat，Cat

17）新建一个流对象，下面哪个选项的代码是错误的？（B）

A. new BufferedWriter(new FileWriter("a.txt"));

B. new BufferedReader(new FileInputStream("a.dat"));

C. new GZIPOutputStream(new FileOutputStream("a.zip"));

D. new ObjectInputStream(new FileInputStream("a.dat"));

18）mysql中想要将选出结果求和的函数是（B）

A. COUNT B. SUM C. AVG D. CONCAT

19）访问修饰符作用范围由大到小是(D)

A. private-default-protected-public

B. public-default-protected-private

C. private-protected-default-public

D. public-protected-default-private

20）程序运行时会检查数组的下标是否越界，如果越界，会出现（C）异常

​ A. NullpointerException

B. ArithmeticExceptioin

C. ArrayIndexOutOfBoundsException

D. SecurityManager

1. 填空题（10\*2）

1）面向对象的三大特性是封装，（继承），（多态）。

2）java中可以通过使用（throw）来声明异常的抛出，可以不捕获的异常类是（RuntimeException），

3）设有数组定义：int a[ ] = { 10 , 20 , 30 , 40 , 50 , 60 , 70}; 则执行以下几个语句后的输出结果是（160）。

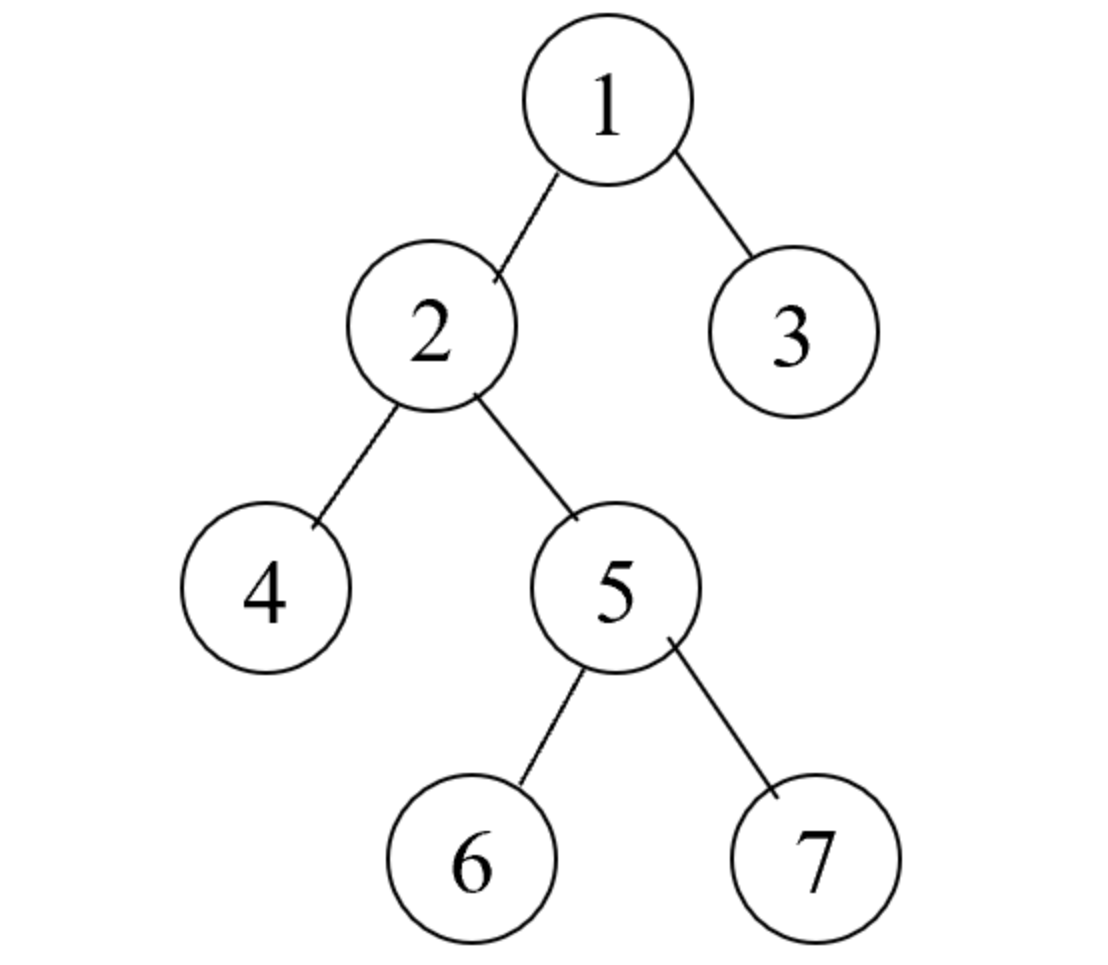
int s = 0 ;

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

if ( i % 2 = = 0 ) s += a[i] ;

System.out.println( s );

4）下图中的树，前序遍历（1245673），后序遍历（467531）。



5）多态是指 子类覆盖了父类的方法 ，在Java中有两种多态，一种是使用方法的（重载）实现多态，另一种是使用方法的（重写）实现多态。

6）设 int x=1 , y=2 , z=3 ， 则表达式 y+=(z--)/(++x) 的值是（3）

7）在 Java 程序中有如下代码，则输出结果为（45）。

Integer x=23456;

System.out.println(x.toString().substring(2,4));

8）表达式'A' + 12的值为（77）

9）mybatis可以通过resultMap的子元素（association）进行一对一关联查询，通过子元素（collection）进行多对多关联查询。

10）SpringMvc中，使用注解（@ResponseBody）可以将接口数据返回给客户端。

1. 简答题（5\*4）
2. 创建线程有哪几种方式？并详细解释

①. 继承Thread类创建线程类

* 定义Thread类的子类，并重写该类的run方法，该run方法的方法体就代表了线程要完成的任务。因此把run()方法称为执行体。
* 创建Thread子类的实例，即创建了线程对象。
* 调用线程对象的start()方法来启动该线程。

②. 通过Runnable接口创建线程类

* 定义runnable接口的实现类，并重写该接口的run()方法，该run()方法的方法体同样是该线程的线程执行体。
* 创建 Runnable实现类的实例，并依此实例作为Thread的target来创建Thread对象，该Thread对象才是真正的线程对象。
* 调用线程对象的start()方法来启动该线程。

③. 通过Callable和Future创建线程

* 创建Callable接口的实现类，并实现call()方法，该call()方法将作为线程执行体，并且有返回值。
* 创建Callable实现类的实例，使用FutureTask类来包装Callable对象，该FutureTask对象封装了该Callable对象的call()方法的返回值。
* 使用FutureTask对象作为Thread对象的target创建并启动新线程。
* 调用FutureTask对象的get()方法来获得子线程执行结束后的返回值。

1. 什么是 spring cloud？

从字面理解，Spring Cloud 就是致力于分布式系统、云服务的框架。

Spring Cloud 是整个 Spring 家族中新的成员，是最近云服务火爆的必然产物。

Spring Cloud 为开发人员提供了快速构建分布式系统中一些常见模式的工具，例如：

* 配置管理
* 服务注册与发现
* 断路器
* 智能路由
* 服务间调用
* 负载均衡
* 微代理
* 控制总线
* 一次性令牌
* 全局锁
* 领导选举
* 分布式会话
* 集群状态
* 分布式消息
* ……

1. rabbitmq 的消息是怎么发送的？

首先客户端必须连接到 RabbitMQ 服务器才能发布和消费消息，客户端和 rabbit server 之间会创建一个 tcp 连接，一旦 tcp 打开并通过了认证（认证就是你发送给 rabbit 服务器的用户名和密码），你的客户端和 RabbitMQ 就创建了一条 amqp 信道（channel），信道是创建在“真实” tcp 上的虚拟连接，amqp 命令都是通过信道发送出去的，每个信道都会有一个唯一的 id，不论是发布消息，订阅队列都是通过这个信道完成的。

1. redis 为什么是单线程的？

因为 cpu 不是 Redis 的瓶颈，Redis 的瓶颈最有可能是机器内存或者网络带宽。既然单线程容易实现，而且 cpu 又不会成为瓶颈，那就顺理成章地采用单线程的方案了。

关于 Redis 的性能，官方网站也有，普通笔记本轻松处理每秒几十万的请求。

而且单线程并不代表就慢 nginx 和 nodejs 也都是高性能单线程的代表。

1. 编程题（20\*1）

一个射击运动员打靶,靶一共有10环,连开10枪打中90环的可能行有多少种?

思路：使用递归思想

首先考虑到一共要打十枪。则可以分三种情况来考虑：

1.如果当前超过了十枪或者积分超过了给定值，则返回

2.如果当前为第十枪，判断最后一枪的积分是否有可能达到给定值（0~10），如果可以，可能值加1，然后返回；否则直接返回。

3.其他情况下，继续递归

private static int sum = 0;

private static int SCORE = 90;

public static void main(String[] args) {

compute(10, 0);

System.out.println(sum);

}

public static void compute(int num, int scores) {

if (num <= 0 || scores > SCORE) {

return;

}

if (num == 1) {

if (scores + 10 >= SCORE) {

sum++;

return;

}

}

for (int i = 0; i <= 10; i++) {

compute(num - 1, scores + i);

}

}