**Dian团队2021秋季招新笔试** [](af://n0)

|  |
| --- |
| **说明部分** |
| **时间安排**  [hr@dian.org.cn](mailto:hr@dian.org.cn)  笔试时间为 2021年9月27日晚 19： 30-22： 30  **提交方式**  将**答卷（pdf文件格式）** 和自己的**最终简历（pdf文件格式）** 压缩打包，发送至邮箱 压缩包命名格式为 **选择的题目-学号-姓名**，例如： 2-U202166666-张三  同时请在答卷**醒目的位置**书写自己的**个人信息**  注意：请**按规定命名**，未按规定命名提交的试题**不予批改！**  **试题部分** |

**试题说明** [](af://n11)

1. 从笔试题1、 2、 3、 4、 5中**任选一道**作答，在**完整解答一题**后可以选择写其他题（请先保证选择的 题目的**作答完整**，多选的题仅仅可作为参考项）

2. **个人独立完成**，**允许查阅资料**，如有答案需要大量复制资料内容，请**附上答案来源** （网址或书籍页 数） ,避免被误认为抄袭

3. **严禁抄袭** （一经发现答案雷同，一同**取消资格**并会被**拉入团队招新黑名单**）

4. 所有年级统一试题。为了体现区分度，题目设置了阶梯性难度，对不同年级**评价标准会有不同**，低 年级零基础的同学尽力而为，尽可能体现自己在整个过程的**思考探索**

5. 若对试题有任何疑问（如题目有误、题意不明确等问题），可联系QQ： **1011137273**（2021秋招 招新群群主）进行咨询

6. 笔试结果后续将**通过手机短信**进行通知发送

**试题内容** [](af://n25)

**笔试题1** [](af://n26)

考察方向： **设计模式、面向对象**

关键词： **UML类图**

**试题1** [](af://n29)

请查询相关资料，回答下列问题。

注意： **”[\*\*\*]“** 内为你的回答中**理应涉及**到并简要解释的内容(也提示了你可能需要搜索的内容)

1. 大致了解面向对象思想，了解任何支持面向对象语言中的类、对象等相关概念**(此题无须作答)**

2. 了解类的继承、函数重写的概念，谈谈你对虚函数（抽象函数）重写的认识**[**函数重写的使用场 景、重写的作用与意义**]**

3. 学习UML类图的相关概念**(此题无须作答)**

**试题2** [](af://n39)

请根据你学习到的类、对象、 UML等的相关知识，回答下列题目。（**画图前先阅读附录**中有关该题的一 些简单约定）

1. 查询资料，了解设计模式中的策略模式。 **[**设计模式、策略模式**]**

现有情景：某商场打折，现有3种打折方式A、 B、 C，可用DisCountA， DisCountB， DisCountC 表示，针对顾客购买货物的总价，会选择不同的打折方式。

大致伪代码如下，请根据代码中表示的关系与逻辑画出UML图：

|  |
| --- |
| //示例代码，不用关系细节，仅表示类关系与大致逻辑  //DisCount父类，提供函数Count接口，供子类重写  Class DisCount  {  public:  virtual int Count(int price) = 0;  }  //DisCountA类，继承DisCount，并重写Count函数，实现对应的打折策略  Class DisCountA: public DisCount  {  public:  int Count(int price);  }  //DisCountB类，继承DisCount，并重写Count函数，实现对应的打折策略  Class DisCountB: public DisCount  {  public:  int Count(int price);  }  //DisCountC类，继承DisCount，并重写Count函数，实现对应的打折策略  Class DisCountC: public DisCount  {  public:  int Count(int price);  }  //伪代码  //假设输入的price值一定在1-10000之间  int CountNewPrice(int price)  {  DisCount\* pDisCount = nullptr;//创建一个DisCount对象指针，可以理解为它指向一个 DisCount对象，但此处指向空  if(price < 1000)  {  pDisCount = new DisCountA;  }  else if(price < 5000 && price >=1000)  {  pDisCount = new DisCountB;  }  else  {  pDisCount = new DisCountC;  }  return pDisCount->Count(price);  } |

2. 假设你正在使用C/C++语言学习服务器开发，在了解到一般服务器都具有日志系统后，作为C++程 序员，你又开始折腾上了——造轮子，你决定写一个自己的日志系统（采用面向对象的思想），在 你的程序里，该日志系统应该是一系列类的组合。请用 UML类图来表达出你的设计。

规定**至少**有如下3个类，并且在设计时尽量使得每个类简洁，功能单一化：

**Logger**：日志器，作为接口，用于让其他组件使用。

**Appender**: 输出器，用于向固定的文件输出指定内容，一个输出器对应一个文件。

**Formatter**：格式器，用于将输出进来的日志信息按照事先规定好的格式化输出方式添加相应信 息进行格式化。

**说明：**

仅设计， **无需实现**。但设计过程请**考虑到实现的难易程度**。

每**一个要求**完成后，请**保存**UML图并当作答卷的一部分进行提交，再进行后续的更改。

**要求：**

1. 为了方便服务器其他组件进行使用，我们会希望有一个 Logger 类，以如下的方式进行调 用：

2.

3.

|  |
| --- |
| //简单调用逻辑代码，非完整代码  //定义日志级别枚举  enum Level  {  INFO,  DEBUG,  WARNING,  ERROR,  FATAL  };  Logger logger; //定义一个logger对象  //若干操作，配置logger,此处略去  ...  string logInfo("log a log");  logger.log(DEBUG,logInfo); //以DEBUG的日志级别，记录了logInfo日志(一个字符 串) |

因为日志信息除了你需要打印的LogInfo以外，希望可以以固定的格式进行输出，并可以添加 相应的信息，比如时间信息等，这个过程我们称之为格式化输出，我们希望能以如下方式进 行设置。

|  |
| --- |
| //设置格式化方式  logger.SetFromatter("[%Y-%M-%D][%T][%m]");//此处 %加一个字母表示格式化字符， 在函数中会使用相对应的值来替代该内容，无需过多关注  //此处格式化的效果大致如此 [2021-09-27][22:29:59][55555题目好难]，其中%m表示你 输入的LogInfo(一个字符串) |

我们同时希望能够一次日志可以打印到多处，所以希望对于一个logger，他可以向多处打 印，且各处打印希望有不同的格式化输出。

|  |
| --- |
| //添加打印文件名  logger.AddAppender("/home/Dian/logFile/20210927.log");//此处传入的参数为 string |

4. 完善功能。在完成上述3题要求后，你现在已经可以如下使用你的日志系统了。 **请先保存好你**

**当前的UML图作为1-3要求的答案提交！**

|  |
| --- |
| //日志系统的测试  int main(int argc,char\* argv[])  {  Logger logger;//创建一个Logger对象  logger.SetFromatter("[%Y-%M-%D %T][%m]");//设置该Logger对象的格式化方式 logger.AddAppender("/home/Dian/logFile/20210927.log");//添加一个输出器 logger.AddAppender("/home/Dian/BackUp/logFile/20210927.log");//再添  加一个输出器  while(true)  {  logger.log(DEBUG,"test log system\n");//输出日志  Sleep(1000);  }  } |

但为了你的日志系统的易用性，你还有很多路要走，试着完成下列一些功能。

1. 比如你日志器中的Appender无法删除，你需要建立路径与appender的对应关系，保存 在适当位置，可以采用如下方式删除某个Appender



//删除 输出器

logger.DelAppender("/home/Dian/logFile/20210927.log");

2. 目前你的所有Appender共用一个Fromatter，若你想定制每个输出器的格式化方式，你 该如何设计？

3. 发挥你自己的创意，添加任何你觉得可以使用的上的功能，并在你的UML设计图中画出 来（配上适当的文字说明）

**笔试题2**

考察方向： **C、嵌入式**

关键词： **IIC协议**

**试题1**

通过搜索引擎查找**IIC协议**相关介绍，简单介绍一下IIC协议，从协议的**总线描述**， **时序图**，以及**设备主** **从关系图**方面描述一下自己的理解。

**试题2**

阅读附件TMP75的datasheet，回答以下问题

1. TMP75的**用途**是什么？

2. 其**测量的范围**是？ **精度**是多少？ **测量所需时间**有多少？

3. TMP内部有几个寄存器，其**作用**分别是什么？

**试题3**

编码题，请根据TMP75的数据手册以及自己了解的IIC协议内容，解答以下问题。

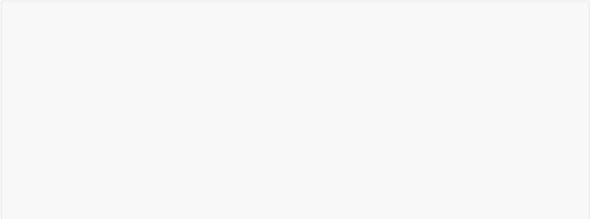
假设你现在有一块TMP75且已经正确连接到上位机上。

1. 找出使用IIC协议向TMP75寄存器写数据的**时序图**，以及使用IIC协议从TMP75寄存器读取数据的**时** **序图**。

2. 假设给出以下接口，请使用这些接口写出初始化TMP75并且从TMP75中读取测量到的数据的代 码，补充在main函数中（仅可能地加上注释）

|  |
| --- |
| //接口说明  设置IO口电平  pin: 目标IO口  level: 目标电平，高电平或低电平  延迟一定时间 us: 延迟的微秒数  typedef enum {  SDA,  SCL  } pinType; // IO口种类定义  typedef enum {  HIGH,  LOW  } electricalLevel; // 电平高低，HIGH为高电平，LOW为低电平  /\*\*  \* @brief  \* @param  \* @param  \*/  void writePin(pinType pin, electricalLevel level);  /\*\*  \* @brief 读取目标电平  \* @param pin: 目标IO口  \* @return 返回读取到的IO口电平，高电平或低电平 \*/  electricalLevel readPin(pinType pin);  /\*\*  \* @brief  \* @param  \*/  void delayTime(int us); |

接口的一些使用说明以及main函数如下，请补充main函数内容，完成初始化TMP75并且从TMP75中读 取测量到的数据的功能。



typedef enum {

SDA,

SCL

} pinType; // IO口种类定义

typedef enum {

HIGH,

LOW

} electricalLevel; // 电平高低，HIGH为高电平，LOW为低电平

void writePin(pinType pin, electricalLevel level); electricalLevel readPin(pinType pin);