第四届软件大赛
比赛规则与内容说明

组别变化

- 4个组别改为6个组别
- 原来的本科组细分为本科A组、本科B组
- 一本院校只能报本科A组，其它本科院校可自行选择A组或B组。

c与c++的选择

- 题面上不会考核c++或面向对象的具体知识
- 仅仅是**编程大题**允许使用c++解答
- 其它题目中不会出现c++知识
- 虽然推荐使用vc工具，但不能使用非标准的c++类库，因而不能使用微软的MFC或ATL类库，不能使用windows API

C++涵盖范围

- 不会在给出的代码中含有类、对象等c++知识
- 不会在题目的描述中强制使用c++
- 对大题的解答允许使用c++编码

- 允许范围：ANSI C++标准
  - 类，继承，多态….
  - 允许使用STL类库
  - 不允许使用 MFC类库
  - 不允许使用 ATL类库

环境变化（1）

- Java环境没有变化
- 只允许JDK1.5，6.0以上特性禁止
- IDE： JCreator2.0， Eclipse helios release2(不含JavaEE特性)
- 请检查IDE连接的JDK是否符合要求
- 检查eclipse是否开启了泛型功能
- 无论用何种工具，提交的结果：
  - 只有一个文件
  - 不包含任何工程配置文件

环境变化（2）

- C语言IDE：borland c++3.1 简化版，VC简化版
- Borland c++ 3.1 与TC2.0 类似 支持c++
  - 不支持 STL
  - 需要STL特性只能用VC

- VC为简化版
  - 没有安装MFC库，ATL库
  - 不可以使用CString等MFC类库
  - 不可以使用CFile等类进行文件操作
  - 不支持界面编程（规则中不允许调用Windows特定的API）

题型变化

- 编程大题没有变化
  - 仅仅是c/c++组允许使用ANSI C++ 特性
- 填空题变化
  - 代码填空 → 代码填空 + 结果填空
  - 代码填空:
    - 读懂已知程序的逻辑，合理填空。
    - 难度在于分析逻辑
  - 结果填空:
    - 不限制实现的手段（可以猜测或手工计算），只要给出结果就可以。
    - 举例：1000! 中含有多少个数字2?

必须的基础知识范围

- 大赛题目的设计本着尽可能需要最少的基础知识的原则
- 比的是组织逻辑的能力，不是对某个偏僻特性的记忆
- 所需的基础知识是绝大多数学校教材内容的交集

- A组的少量题目可能会超越课本范围（后面列出）

必须的领域知识

- 大赛题目的设计本着需要尽可能少的领域知识的原则
- 所有特殊领域知识，一定会在题目中详细描述，一定会有例如…的说明语句，以免选手误解。

- 例外：数学领域
- 数学常识性知识不在题目中详述！
  - 高中以内的数学知识
  - 算数：素数，整出，余数，求模，不定方程 …
  - 代数：函数，方程，多项式，…
  - 解析几何：笛卡尔坐标系，点到直线的举例，极坐标，…
  - 复数：模，夹角，矢量的合成和分解

Java组别基本

- 基础：变量，操作符，选择，循环，递归
- 面向对象：类，对象，引用，构造方法，参数传递，this引用，static，继承，多态，接口，内部类，匿名类
- 异常与保护
- I/O：读写文件，File类，文本操作与二进制操作（字节流，字符流）
- 多线程：线程的排斥（synchronized），协作（wait, notify）
- 网络：只限于Socket通信

- 不会出现：
- AWT，Swing界面类的编程或填空问题
- JSP, Servlet, HTML, CSS,XML, JavaScript 等web编程相关
- Struts, Spring等开源框架
- JavaEE 规范，容器（例如: JNDI， javaBean等）
- JDBC, SQL 等数据库编程相关内容

Java组

- Java本科B组增加
  - 对多种数据结构的灵活运用
- Java本科A组增加
  - 设计模式，反射，XML，多核与并发，测试理论，Swing界面
  - 仅限于少量题目中可能涉及

c/c++组基本

- 基本：变量，运算符，选择，循环，数组，指针，递归
- I/O：读写文件，标准输入输出，文件属性

- 不会出现：
- 含有窗口的Windows界面编程
- 多线程
- 网络编程、Web应用
- 数据库编程
- 调用底层中断或硬件相关的编程
- 其它一定需要非ANSI C标准调用的编程

c/c++组

- c/c++本科B组增加
  - 数据结构、函数指针、位运算
- c/c++本科A组增加
  - 函数模板、复杂宏、汇编知识
  - 仅限于少量题目中可能涉及

数据结构与算法

- 本科组《数据结构》教材为准
- 可能直接引用其中的术语，不再做解释。比如：题面中可能出现：这是一个平衡二叉树，至于什么是平衡二叉树可以不解释，当作是选手清楚的基础知识。

- 高职高专组，不会在题面中出现《数据结构》课程中才有的特定术语；如实在无法避开，会给出那个概念的详细解释。
- 但允许选手使用任何《数据结构》中的技巧或实现。

- C++ 允许使用 STL
- Java 允许使用 JDK1.5范围内的API

算法

- 穷举法(暴力破解)
- 回溯法（试探，返回，试探，返回…）
- 分治法
- 动态规划

- 算法不是死的，可以有各种灵活的“杂合”或“变种”
- 程序填空题中可能含有某个算法的思想，需要能读懂
- 专科组除了“穷举法”，其它不会在题面代码中出现

关于递归

- 递归是解决复杂问题的重要手段
- 需要掌握递归与循环的转化关系

- 有些结果填空题或大题完全放弃递归会很吃力
- 取球问题举例

评分方法

- 阅卷方式：人工 + 程序辅助

- 代码填空题
- 与标准答案一致，得分
- 不一致的带入测试程序，结果正确得分

- 选手理解错误：不需要填写其它，只填写缺少的部分
- 选手粗心：分号已经存在了，中西文符号问题
- 粗心会按统一标准扣分，将来全自动机器阅卷可能完全不给分

- C代码填空举例

评分方法

- 编程大题
- 运行结果的正确性比重 >90%
  - 如果输入结果不正确，评审时一般不会去分析其错误的原因
  - 如果编译有问题，会去排除环境差异的干扰，若没有按要求提交代码，而运行结果正确，适当扣分

- 存在问题
  - 测试用例与题面举例不同，不能用printf System.out.println 蒙混
  - 使用标准输入、输出。 便于重定向测试。
  - 严格按题目要求，不要画蛇添足。输出的内容不要有多余的东西。
    - 将来用全自动机器阅卷会判负
  - 思维要严密。边界条件判断不足，引发异常，会酌情扣分。
  - 大数据规模。算法设计不当会导致溢出或速度不可忍受。

评分标准

- 大题测试通过，如何比拼?
- 依运行时间加分（体现算法的效率）
- 如果运行时间相仿?
  - 依代码的规范性、可读性、可维护性加分

题目难度

- 坚持原创性，押题不容易。
- 难度略低于ACM
  - 有些题目受NOI， IOI， POJ， ACM，TopCode等知名赛事题目的启发
- 注重普及性，让尽可能多的选手参赛本身可以获得启发或收获
  - 大赛的目标之一：传播知识
- 难度高于期末考试，更重要的是注重了趣味性。
  - 举例-魔方问题
- 不同于纯粹的语法考试，注重在工程应用领域的意义。
  - 有些题目原型是高校或研究单位的论文成果。
  - 举例-图像面积计数问题

如何备战

- 基础知识扎实
  - 真正理解，彻底理解，不是一知半解!
- 主要是培养逻辑能力，可以通过各种题目练习
  - 数学练习很重要，最有成效!
  - 例如：欧拉计划 网站
  - 国外教材的习题很有挑战性、开放性
  - 《java大学教程》 《c++大学教程》

题目素材或背景

- 数学素材，最普遍
- 串的各种变换
  - java允许使用正则，可能会事半功倍
- 文件内容的处理
  - 文本文件转换格式，比较，搜索等
  - 二进制文件提取某种信息，某种映射
- 仿真问题
  - 类似电梯调用，餐厅调用等。。
  - 各种概率问题
- 求最优解问题
- 博弈问题