C Project

- **Presenter:**
  Wei.Xue@Sun.COM
  Sun ERI

Agenda

- 代码行统计工具
- 设计小游戏
- 自然语言处理 - 汉字拼音标注
- 项目要求
- 考核方式

代码行统计工具

- 问题描述功能需求
  - 对指定的目录下的源代码文件，（如果有头文件包括头文件）进行统计并输出结果
  - 能统计一个给定文件的代码行
  - 界面友好
    - 如命令行运行时后面加参数，没有参数能提示输入参数
    - 列出对每个文件的统计结果以及总的统计结果（百分比）

代码行统计工具

- 要求
  - 需要完成对 C， C++， java 语言的代码统计
    - 可通过配置参数定制被统计源文件的扩展后缀名
  - 统计内容包括
    - 物理代码行数
    - 逻辑代码行数
    - 总行数
    - 注释行数
    - 空行数
    - 无效代码行数

代码行统计工具

- 说明及难点
  - 空行
    - 代码间的空行属于空行，注释间的空行也属于空行
  - 逻辑行与物理行
    -

    A=5; B=6;

    逻辑行 =2 ， 物理行 =1

    -

    a=5+b\

    +c;

    逻辑行 =1 ， 物理行 =2

代码行统计工具

- 说明及难点
  - 注释行
    - C， C++:

      /*... ...*/

      //...

    - Java

      包括 C++ 的注释方法，并且还有 javadoc

      /** ..... */

代码行统计工具

- 说明及难点
  - 注释行常见难点 (1)
    - "....../*....*/....." 不属于注释行
    - "....../*....\
      ....*/....." 属于代码行
    - "....\
      /*....\
      ....*/" 属于代码行

代码行统计工具

- 说明及难点
  - 注释行常见难点 (2)
    - "……\"\
      /*……\
      ……*/……" 属于代码行
    - "……\
      //……" 属于代码行
    - 代码行中间存在 /*……*/ 情况， 只记代码行一行

代码行统计工具

- 说明及难点
  - 代码行无效的代码行
    - 除去注释行和空行都是代码行
    - 考虑条件编译行

```
#if 0
a=b;
#endif
无效的代码行
```

代码行统计工具

Q&A

小游戏 - 俄罗斯方块 (双人版)

- 需求描述
  - 基本描述
    - 随机给出不同的形状（长条形、 Z 字形、反 Z 形、田字形、 7 字形、反 7 形、 T 字型）下落填充给定的区域，若填满一条便消掉，记分，当达到一定的分数时，过关，设置五关，每关方块下落的速度不同，若在游戏中各形状填满了给定区域，为输者。
    - 支持双人版。

小游戏 - 俄罗斯方块 (双人版)

- 需求描述
  - > 游戏界面需求 :

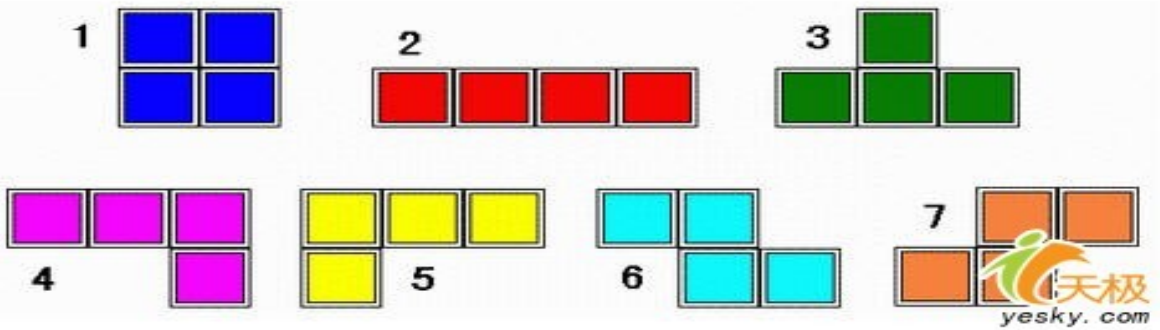
良好的用户界面，有关数显示和分数显示。让方块在一定的区域内运动和变形 , 该区域用一种颜色表明 , 既用一种颜色作为背景 , 最好设为黑色。还需用另一种颜色把黑色围起来 , 宽度适中，要实现美感。

要求显示下一个方块，当前分数，记录等信息。

小游戏 - 俄罗斯方块 (双人版)

- 需求描述
  - > 游戏形状 (方块) 需求 :

绘制七种常见的基本图形（田字形、长条形、 T 字型、 7 字形、反 7 形、 Z 字形、反 Z 形），各个方块要能实现它的变形，可设为顺时针或逆时针变形，一般为逆时针。



小游戏 - 俄罗斯方块 (双人版)

- 需求描述
  - > 键盘处理事件 :

方块下落时，可通过键盘方向键（上、下、左、右键）对该方块进行向上 (变形), 向下（加速）、向左、向右移动。

  - > 显示需求 :

当不同的方块填满一行时可以消行，剩余方块向下移动并统计分数。当达到一定的分数时过关。设置五关，每关方块下落的速度不同。

小游戏 - 俄罗斯方块 (双人版)

- 需求描述
  - 计分规则 :
    - 一次消去一行记 1 分,
    - 一次消去两行得 3 分
    - 一次消去三行得 6 分,
    - 一次消去四行得 10 分
  - 保留分数历史排名
    - 可保留前五名
    - 可进行历史记录的重新清空设置

小游戏 - 俄罗斯方块 (双人版)

- 需求描述
  - > 双人游戏规则 :

  假设玩家 A,B 同时玩，如果 A 连续消去 4 行，而 B 没有消去一行，则在 B 中随即产生一行其中的空格数和位置随即产生。

  一方先满的为输

小游戏 - 俄罗斯方块 (双人版)

- 说明及提示
  - GUI 方法 :
    - 纯文本
    - curses
    - Xlib
    - gtk

小游戏 - 俄罗斯方块 (双人版)

- 说明及提示
  - 进程间的通信
    - 在双人版考据进程间的通信同步等问题

小游戏 - 俄罗斯方块 (双人版)

- 说明及提示
  - 通过命令行参数的制定切换不同的启动模式
    -s 启动单人版
    -w 启动双人版

小游戏 - 俄罗斯方块 (双人版)

- 说明及提示
  - 通过命令行参数的制定切换不同的启动模式
    -s 启动单人版
    -w 启动双人版

小游戏 - 俄罗斯方块 (双人版)

- Q&A

拼音标注

- 需求描述

  对输入的汉字输出其对应的拼音标注。采用格式：拼音声调。例如：

  输入为：拼音标注

  则输出的注音应该是：

  pin1yin1biao1zhu4

  对应轻声的音调为 5

拼音标注

- 难点分析

  汉字有限个数，在字典中查询读音。但是：

  多音字

  多音词

拼音标注

- 解决方法 - 罗列词
  - 首先为每个多音字确定一个常用音，然后尽可能将包含多音字的非常用音的词罗列在一个词典中。在字 - 音转换时，如果查到匹配的词就根据词典确定读音，如果没有查到，就采取缺省读音。
  - 问题
    - 许多常用多音字如“为”、“长”等的非缺省读音的使用频度也很高，而且很难用词典穷尽。
    - 多音词需要根据上下文确定读音，如“播种”、“背着”，“朝阳”

拼音标注

- 解决方法 - 基于语料库的统计模型分析
  - 预备概念
    - 语料库
    - 字典
    - 最大正向分词
  - 现有的一些模型
    - N-gram
    - 最大熵
    - HMM

拼音标注

- 其他解决方法
  - 在线工具：
    - http://py.kdd.cc/
    - http://www.pinyinannotator.com/
- 迄今为止没有完全解决

拼音标注

- Q&A

项目要求

- 实现要求
- 文档
- 编码规范

项目要求

- 实现要求
  - 将常用可重用的函数实现为一个动态连接库。学习动态连接库的编写和使用
  - 完成所有要求文档
  - 严格遵守编码规范

项目要求

- 文档
  - 设计书
    - 项目的分析
    - 设计方案（模块化）
      - 设计方案中应包括难点问题的解决思路和关键的数据结构和算法
    - 测试计划
    - 组员任务分配
    - 工作计划
  - 项目日志
  - 用户手册，帮助

项目要求

- 编码规范
  - 目的

为了保证在软件开发过程中，全体成员的代码风格一致，便于维护，提高软件产品的质量和保持开发产品的延续性，特制定本编码规范。

项目要求

- 编码规范
  - 文件名
    - 代码文件使用下列扩展名：

文件类型	扩展名
C 源文件	.c
C++ 源文件	.cpp,.cxx
C/C++ 头文件	.h

项目要求

- 编码规范
  - 文件名
    - 常用文件名：
      - Makefile ： Makefile 的首选名称
      - RELEASE ：程序发行说明，描述各个版本的主要更改、问题解决状况、尚未解决的问题等等．首选文件名．
      - README ：总结项目目录的内容。
      - ChangeLog ：代码修改记录，详细描述对每个源文件所做的修改，最好精确到函数级别。结构上，可采用如下形式：

        ChangeLog

        + 日更改记录

        + 文件修改记录

项目要求

- 编码规范
  - 文件名
    - 组织形式：

      Project
      +src
      +lib
      +include
      +doc
      README
      ChangeLog
      RELEASE
      其它工程文件如 IDE 相关的项目配置文件，Makefile

项目要求

- 编码规范
  - 代码文件结构
    - 文件头

      版权说明

      公司或组织的名称

      开发人员

      地址（联系方式）

      模块目的 / 功能

      文件版本

      修改日志

项目要求

- 编码规范
  - 代码文件结构
    - 文件头

      版权说明

      公司或组织的名称

      开发人员

      地址（联系方式）

      模块目的 / 功能

      文件版本

      修改日志

项目要求

- 编码规范
  - 代码文件结构
    - C 头文件

      文件头

      版本历史

      #include 区 （可选）

      常量定义

      全局宏定义

      全局数据类型定义

      全局变量定义

      外部引用定义

      全局函数原型定义

项目要求

- 编码规范
  - 代码文件结构
    - C 源文件

      文件头
      版本历史
      #include 区
      #define 区
      宏定义区 (Macros)
      本地数据类型定义区 (Local data types)
      本地变量区 (Local variables)
      本地函数原型 (Local function prototypes)
      局部函数 (Local functions)
      全局函数 (Global functions)

项目要求

- 编码规范
  - 排版
    - 函数、结构、循环、判断等语句都需要采用缩进，缩进请使用 4 个空格
    - 代码行宽度，推荐限制在 80 个字符内
    - 相对独立的程序块之间、变量说明之后必须加空行。示例：

```
void function1()
{
  int var1,var2;

  var1 = 1;
  var2 = 2;
}
```

项目要求

- 编码规范
  - 排版
    - 不允许把多个短语句写在一行中
    - 若函数或过程中的参数较长，则要进行适当的划分
    - If, for, while 等的写法 ,{ 与 if, for, do 在同一行：

```
if (a == b) {
    do something;
}
```

项目要求

- 编码规范
  - > 排版
    - 在两个以上的关键字、变量、常量进行对等操作时，它们之间的操作符之前、之后或者前后要加空格；进行非对等操作时，如果是关系密切的立即操作符（如 ->），后不应加空格。
      - 逗号、分号只在后面加空格。

        int a, b, c;
      - 比较操作符，赋值操作符 "=" 、 "+=", 算术操作符 "+" 、逻辑操作符 "&&", %", "&", 位域操作符 "<<" 、 "^" 等双目操作符的前后加空格。

```
if (current_time >= MAX_TIME_VALUE)
a = b + c;
a *= 2;
a = b ^ 2;
```

项目要求

- 编码规范
  - 排版
    - 空格
      - "!"、"~"、"++"、"--"、"&"(地址运算符) 等单目操作符前后不加空格。

```
*p = 'a';           // 内容操作 "*" 与内容之间
flag = !isEmpty;  // 非操作 "!" 与内容之间
p = &mem;         // 地址操作 "&" 与内容之间
i++;                // "++","--" 与内容之间
```

      - "->"、"." 前后不加空格

```
p->id = pid; // "->" 指针前后不加空格
```

项目要求

- 编码规范
  - 排版
    - 空格
      - if、 for、 while、 switch 等与后面的括号间应加空格，使 if 等关键字更为突出、明显。

        ```
        if (a >= b && c > d)
        ```

      - 针对变量的条件判断语句，

项目要求

- 编码规范
  - 注释
    - 对于函数要给出必要的注释，列出：函数的名称、功能、入口参数、出口参数、返回值、调用说明.

```
/**
 * sample_func – summary this function
 * Parameter:
 *     param1(IN/OUT):      description of param1
 *     param2(IN):      description of param2
 *  return value:
 *      1 -
 *      0 -
 * * Detail descript the usuage of sample_func.
 */
int sample_func(int param1, int param2);
```

项目要求

- 编码规范
  - 注释
    - 注释的格式尽量统一，建议使用 /* ... */
    - 将注释与其上面的代码用空行隔开

```
/* code block one comment */
code block one

/* code block two comment */
code block two
```

    - 每 1 个注释都必须有用

项目要求

- 编码规范
  - 标志符命名
    - 标识符的命名要清晰、明了，有明确含义
    - 命名规范必须与所使用的系统风格保持一致，比如采用 UNIX/linux 的全小写加下划线的风格或大小写混排的方式，不要使用大小写与下划线混排的方式。
    - #define constants:
    - #define macros:
    - typedefs:
    - enum tags:

    所有字符都必须大写

项目要求

- 编码规范
  - 变量
    - 尽可能少定义全局变量
    - 严禁使用未经初始化的变量作为右值
    - 局部变量与全局变量不要同名
    - 当向全局变量传递数据时，要十分小心，防止赋与不合理的值或越界现象发生

项目要求

- 编码规范
  - 函数， 过程
    - 对所调用函数的错误返回码要仔细、全面地处理
    - 编写可重入函数时，应注意局部变量的使用
      - 在函数内部慎用 static 变量
    - 局部变量与全局变量不要同名
    - 编写可重入函数时，若使用全局变量，注意加以保护
    - 在同一项目组内，应明确规定对接口函数参数的合法性检查由函数的调用者还是由接口函数本身负责，缺省是由函数调用者负责
    - 函数的规模尽量限制在 100 行以内

项目要求

- 编码规范
  - 宏
    - 用宏定义表达式时，要使用完备的括号

示例：如下定义的宏都存在一定的风险。

```
#define RECTANGLE_AREA( a, b ) a * b
#define RECTANGLE_AREA( a, b ) (a * b)
#define RECTANGLE_AREA( a, b ) (a) * (b)
```

正确的定义应为：

```
#define RECTANGLE_AREA( a, b ) ((a) * (b))
```

项目要求

- 编码规范
  - > 宏
    - – 使用宏时 , 不允许参数发生变化

示例 : 如下用法可能导致错误。

```
#define SQUARE( a ) ((a) * (a))
int a = 5;
int b;
b = SQUARE( a++ ); //  结果 :a = 7, 即执行了两次增  1 。
```

正确的用法是 :

```
b = SQUARE( a );
a++; //  结果 :a = 6, 即只执行了一次增  1 。
```

项目要求

- 编码规范
  - > 宏
    - 将宏所定义的多条表达式放在 do{}while(0)

例如： #difne DO(a,b) a+b;\
a++;

应用时：

```
if(….)
  DO(a,b); // 产生错误
else
```

解决方法 : #difne DO(a,b) do{a+b;\
a++;}while(0)

项目要求

- 编码规范
  - 可测性
    - 在同一项目组或产品组内，要有一套统一的为集成测试与系统联调准备的调测开关及相应打印函数，并且要有详细的说明。

      示例：以下使用 DEBUG 作为调试开关，输出调试信息

      ```
      #ifdef DEBUG
      #define DPRINTF(s) PRINTF(s)
      #else
      #define DPRINTF(s)
      #endif
      ```

      如果要将调试开关打开，只需要传递给编译器” -DDEBUG” 标记。

项目要求

- 编码规范
  - 可测性
    - 调试信息的格式必须保持一致和有意义：至少要有所在模块名 (或源文件名) 及行号
    - 单元测试代码应在一个子模块
    - 正式发布去点测试部分
    - 测试部分不能影响正常功能和效率

项目要求

- 编码规范
  - 其他
    - 对库函数增加 C++ 支持声明：

示例 :#ifdef __cplusplus
extern "C" {
#endif
int function1();
int function2();
#ifdef __cplusplus
}
#endif

项目要求

- 编码规范
  - 其他
    - 为防止重复包含，请在头文件加上条件 INCLUDE:

示例：

```
#ifndef __LINUX_FILE_H
#define __LINUX_FILE_H
#include <stdio.h>
#include <include/mydefineheader.h>
struct file_struct {
    //....
    //....
};
#endif
```

    - 仅引用你需要的头文件

项目要求

- 编码规范
  - 其他
    - 尽量不要在头文件中暴露数据结构：这样可以用户对你的实现的依赖，也减少了用户的编译时间

```
typedef struct my_State my_State;
my_State *my_open (void);
void      my_close (my_State *L);
```

项目要求

- 编码规范
  - 其他
    - * 函数声明前加 XXX_API 已利于拓展 :

```
#ifndef MY_API
#define MY_API          extern
#endif

MY_API my_State *my_open (void);
```

项目要求

- Q&A

考核方式

主要采用考查方式，每个考核项（如概要设计）均采用百分制，最后加权如下

- 学员总分 = 小组得分*50% + 学员考查得分*30% + 学员考勤分*20%

考核方式

- 考核方式
  - 小组考核方式和评分标准如下：
    - 设计方案： 20％
    - 编码： 30％
      - 代码正确性
      - 代码风格
      - 代码完成
    - 最终结果展示：20%
    - 文档：30％

Thanks

Everyone and Everything Participating on the Network

C Project

Presenter's Name
Wei.Xue@sun.com