# 2 实验报告内容要求

一、效果

二、思路

三、代码

# 3 实验题目

JAVA实现打字游戏

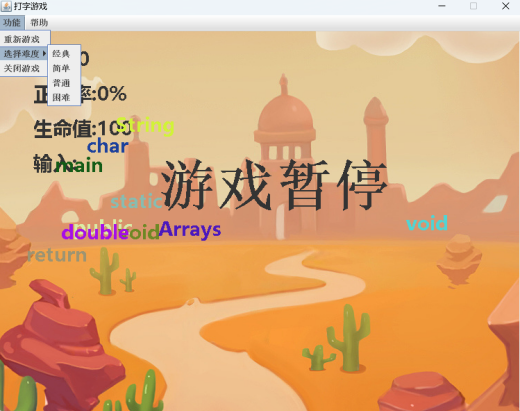
# 4 实验环境

（1）编译、运行环境：JDK8或以上

（2）编辑环境：IDEA

# 5 实验分析和设计思路

**一、效果**

首先我们先看效果，左上角的分数是用来记录我们打对了多少单词。单词是从上面开始往下落。每打对一个单词，分数增加，单词下落的速度玩家可以在功能中选择。  
   


二、思路

一：主函数：程序启动的入口

**public class** Main {  
 **public static void** main(String[] args) {  
 **new** GameMenu();  
 }  
}

二：GameMenu类：搭建游戏界面及菜单

相关函数：

initJFrame()：创建游戏主界面

**private void** initJFrame() {  
 **this**.setTitle(**"打字游戏"**);  
 **this**.add(**panel**);  
 Thread t = **new** Thread(**panel**);  
 t.start();  
 **panel**.addKeyListener(**panel**);  
 **panel**.setFocusable(**true**);  
 **this**.setSize(760, 600);**//760,600  
 this**.setDefaultCloseOperation(3);**//设置关闭模式  
 this**.setLocationRelativeTo((Component) **null**);**//将窗口放到桌面中间**}

initJMenuBar():创建主界面的菜单选项包括游戏难度，关闭游戏，重新游戏及帮助。

**private void** initJMenuBar() {  
 JMenuBar jMenuBar = **new** JMenuBar();  
 **// 创建菜单上的选项（功能，帮助）** JMenu FuncationJMenu = **new** JMenu(**"功能"**);  
 **//添加被点击执行后的功能  
 replayItem**.addActionListener(**this**);  
 **choseItem**.addActionListener(**this**);  
 **closeItem**.addActionListener(**this**);  
 **Help**.addActionListener(**this**);  
 **degree0**.addActionListener(**this**);  
 **degree1**.addActionListener(**this**);  
 **degree2**.addActionListener(**this**);  
 **degree3**.addActionListener(**this**);  
 **//将每一个选项的条目添加到当前选项中  
 choseItem**.add(**degree0**);  
 **choseItem**.add(**degree1**);  
 **choseItem**.add(**degree2**);  
 **choseItem**.add(**degree3**);  
 FuncationJMenu.add(**replayItem**);  
 FuncationJMenu.add(**choseItem**);  
 FuncationJMenu.add(**closeItem**);  
 **HelpJMenu**.add(**Help**);  
 **//将菜单里面的两个功能添加到菜单中** jMenuBar.add(FuncationJMenu);  
 jMenuBar.add(**HelpJMenu**);  
 **this**.setJMenuBar(jMenuBar);  
}

actionPerformed():鼠标监听函数，提供鼠标点击菜单后的执行的功能，如重新游戏，选择难度（经典，简单，普通，困难），游戏提示以及游戏最终结算。

@Override  
**public void** actionPerformed(ActionEvent e) {  
 Object c = e.getSource();  
 **if** (c == **replayItem**) {  
 System.***out***.println(**"重新游戏"**);  
 **this**.setVisible(**false**);  
 **new** GameMenu();  
 }**if** (c == **degree0** ) {  
 **panel**.**f** = **true**;  
 System.***out***.println(**"经典"**);  
 System.***out***.println(**panel**.getSpeed());  
 }  
 **if** (c == **degree1**) {  
 **panel**.**f** = **false**;  
 System.***out***.println(**"简单"**);  
 **panel**.setSpeed(7);  
 }  
 **if** (c == **degree2**) {  
 **panel**.**f** = **false**;  
 System.***out***.println(**"普通"**);  
 **panel**.setSpeed(5);  
 }  
 **if** (c == **degree3**) {  
 **panel**.**f** = **false**;  
 System.***out***.println(**"困难"**);  
 **panel**.setSpeed(2);  
 }  
 **if** (c == **closeItem**) {  
 System.***out***.println(**"关闭游戏"**);  
 System.***out***.println(**"@@@@@@@@@@@@@@@@@@@@@@@@@"**);  
 System.***out***.println(**"@"**);  
 System.***out***.println(**"@ 分数："** + **panel**.getScore());  
 System.***out***.println(**"@ 正确率："** + **panel**.**rate** + **"%"**);  
 System.***out***.println(**"@ 时间："** + (System.*currentTimeMillis*() - **panel**.**StartTime**) / 1000 + **"s"**);  
 System.***out***.println(**"@ 生命值："** + **panel**.**hp**);  
 System.***out***.println(**"@ "**);  
 System.***out***.println(**"@@@@@@@@@@@@@@@@@@@@@@@@@"**);  
 System.*exit*(0);  
 }  
 **if** (c == **Help**) {  
 System.***out***.println(**"提示"**);  
 **panel**.**flag** = **false**;  
 JOptionPane.*showMessageDialog*(**null**,  
 **"你可以选择不同的难度\n按空格表示暂停或开始\n按backspace删除一个字符\n选择功能内的关闭游戏可结算游戏并关闭"**, **"提示"**, JOptionPane.***PLAIN\_MESSAGE***);  
 }  
 **panel**.**flag** = **true**;  
}

GameRun类：游戏运行时的程序

相关函数：

setWords():画布函数，单词出现位置随机，颜色随机

**public void** setWords() {  
 **for** (**int** i = 0; i < 10; i++) {  
 **this**.**xx**[i] = (**int**) (Math.*random*() \* 645);  
 **this**.**yy**[i] = (**int**) (Math.*random*() \* 200);  
 **this**.**colors**[i] = RandomColor();  
 **words**[i] = Words();  
 }  
}

Background():添加背景图片

**public void** background() {  
 **this**.setLayout(**null**);**//取消默认居中放置（针对添加的组件位置）  
 //要和背景图片路径保持一致** ImageIcon im = **new** ImageIcon(**"src\\TypingGameSRC\\BackGround.jpg"**);  
 JLabel jl = **new** JLabel(im);  
 jl.setSize(750, 550);**//750 550  
 this**.add(jl);**//添加背景图片  
 this**.repaint();**//更新界面**}

paint(Grapgics g):画笔函数，设置单词的字体及颜色，将分数，正确率，输入的字母，生命值显示在画布上。

**public void** paint(Graphics g) {  
 **super**.paint(g);  
 **if** (!**flag**) {  
 g.setFont(**new** Font(**"宋体"**, Font.***BOLD***, 80));  
 **if** (!**flag1**)  
 g.drawString(**"游戏结束"**, 230, 250);  
 **else** g.drawString(**"游戏暂停"**, 230, 250);  
 }  
 Font ft = **new** Font(**"微软雅黑"**, 1, 28);  
 g.setFont(ft);  
 g.drawString(**"分数:"** + **this**.**score**, 50, 50);  
 g.drawString(**"正确率:"** + **rate** + **"%"**, 50, 100);  
 g.drawString(**"输入:"** + **s**, 50, 200);  
 g.drawString(**"生命值:"** + **hp**, 50, 150);  
 **for** (**int** i = 0; i < 10; i++) {  
 g.setColor(**this**.**colors**[i]);  
 g.drawString(**this**.**words**[i], **this**.**xx**[i], **this**.**yy**[i]);  
 }  
}

run():多线程方法，使单词不断下移，若单词超过下界，则重新生成新的单词

**public void** run() {  
 **while** (**true**) {  
 **while** (!**flag**) { **//判断是否暂停  
 try** {  
 Thread.*sleep*(100);  
 } **catch** (InterruptedException var2) {  
 var2.printStackTrace();  
 }  
 }  
 **for** (**int** i = 0; i < 10; i++) {  
 **this**.**yy**[i]++;  
 **if** (**this**.**yy**[i] > 560) {  
 **this**.**hp**--;  
 **if** (**hp** == 0) {  
 **flag1** = **false**;  
 **flag** = !**flag**;  
 }  
 **this**.**yy**[i] = 0;  
 }  
 }  
 **try** {  
 Thread.*sleep*((**long**) **this**.**speed** \* 3);  
 } **catch** (InterruptedException var2) {  
 var2.printStackTrace();  
 }  
 **this**.repaint();  
 }  
}

keyPressed():键盘监听函数，如果输入的字母与画布上的某个单词的字母匹配，则该单词会闪烁，当输入的字母构成画布上的某个单词时，该单词会被消除；如果按下空格，则游戏暂停；可以按backspace删除一个已经输入的字母。

@Override  
**public void** keyPressed(KeyEvent e) {  
 Random p = **new** Random();  
 **if** (e.getKeyCode() == 8 && **s**.length() != 0)**//刪除1个字母（当玩家按下backspace时）** {  
 **s** = **s**.substring(0, **s**.length() - 1);  
 }  
 **if** (e.getKeyCode() == 32) {  
 **if** (**flag**) System.***out***.println(**"暂停"**);  
 **if** (!**flag**) System.***out***.println(**"开始"**);  
 **flag** = !**flag**;  
 **this**.repaint();  
 }  
 **if** (Character.*isLetter*(e.getKeyChar())) {  
 **this**.**LetterNumber**++;  
 **s** += e.getKeyChar();  
 }  
 **if** (**s**.length() == 2) { **//当玩家每次输入2个字母后，开始遍历，找到玩家想要删除的单词（这里不是很严谨）  
 for** (**int** i = 0; i < **words**.**length**; i++) {  
 String t = **words**[i].substring(0, 2);  
 **if** (t.equals(**s**)) {  
 **target** = **words**[i];  
 **targetIndex** = i;  
 }  
 }  
 }  
 **for** (**int** i = 0; i < **words**.**length**; i++) {**//遍历单词，找到最靠下的单词  
 if** (**words**[**targetIndex**].equals(**words**[i])) {  
 **if** (**this**.**yy**[**targetIndex**] < **this**.**yy**[i]) {  
 **targetIndex** = i;  
 }  
 **if** (**yy**[i] < **yy**[(i + 1) % 9]) **lowyy** = **yy**[(i + 1) % 9];  
 }  
 }  
 **if** (**s**.length() >= 2)**//当前正在聚焦的单词发生闪烁（当玩家输入一个单词的两个字母后）** {  
 **colors**[**targetIndex**] = **new** Color(p.nextInt(255), p.nextInt(255), p.nextInt(255));  
 }  
 **if** (**lowyy** >= 600) {**//判断是否有单词已经到达最低点  
 hp**--;  
 }  
 **if** (**s**.equals(**target**) && **s**.length() != 0) { **//如果用户输对一个单词  
 if**(**f**){  
 **speed** = **speed** - 0.1;  
 }  
 Voice();  
 System.***out***.println(**s**);  
 **this**.**xx**[**targetIndex**] = (**int**) (Math.*random*() \* 670.0D);  
 **this**.**yy**[**targetIndex**] = 0;  
 **this**.**words**[**targetIndex**] = Words();  
 **this**.**colors**[**targetIndex**] = RandomColor();  
 **if** (**targetIndex** + 1 > 9) **targetIndex** %= **words**.**length** - 1;  
 **this**.**score**++;  
 **this**.**TrueLetter** += **s**.length();  
 **if** (**LetterNumber** != 0) **this**.**rate** = (**int**) ((**this**.**TrueLetter**) / **this**.**LetterNumber** \* 100);  
 **s** = **""**;  
 }  
 **this**.repaint();  
}

Voice():每消除一个单词会产生音效。

**void** Voice(){  
 **final** String f=**"src\\TypingGameSRC\\voice.aif"**;  
 Clip c=**null**;  
 **try** {  
 c= AudioSystem.*getClip*();  
 c.open(AudioSystem.*getAudioInputStream*(**new** File(f)));  
 c.loop(0);  
 }  
 **catch**(Exception ex)  
 {  
 }  
}

# 6 流程分析和类图结构

建立游戏界面类→建立游戏运行类→主函数启动游戏

类的继承，接口的实现：

·JFrame类：面板组件，顶层容器类

·JPanel类：面板组件，非顶层容器类

·ActionListener：鼠标监听接口

·Runnable：实现多线程接口

·KeyListener：键盘监听接口

IMG_256

IMG_256