设计文档

1. 题目

完成一个猜数字的GUI游戏。

开始游戏后，产生一个没有重复数字的4位随机数，用户每猜一个数字，显示出“完全猜中的数字个数”和“猜中数字但位置错误的数字个数”，比如nAmB，数字n表示猜中的位置正确的数字个数，数字m表示数字正确而位置不对的数字个数。例如，正确答案为5234，如果用户猜5346，则显示：1A2B，数字1表示数字5及其位置猜对了，数字3和4这两个数字猜对了，但是位置没对，记为2B。然后，用户根据游戏提示的信息继续猜，直到猜中为止。同时设计规则，根据猜中的次数计算积分，并可以显示不同用户的排行榜。

1. 功能分析
2. 猜数字游戏的基本算法，能够顺利完成游戏
3. 可以计算积分，并记录不同用户的名字，选择排序并输出排行

3.GUI界面

开始游戏界面，包含开始、排名、退出等选项

猜数字游戏界面，包含输入名字，输入数字，开始猜测等

排行界面，输出排行、名字、分数

选项表，支持建立新用户或猜新的数字

1. 设计思路

Run类 开始运行的总体类，包含分支游戏面板start,game,rank

Start类 游戏开始界面，包含按钮和欢迎界面

Game类 猜数字游戏界面 包含获取用户名，获取猜测数，开始按钮，nAmB提示，分数显示等 有client数组，用户数量等属性

Rank类 排行界面 用选择排序比较分数进行排名

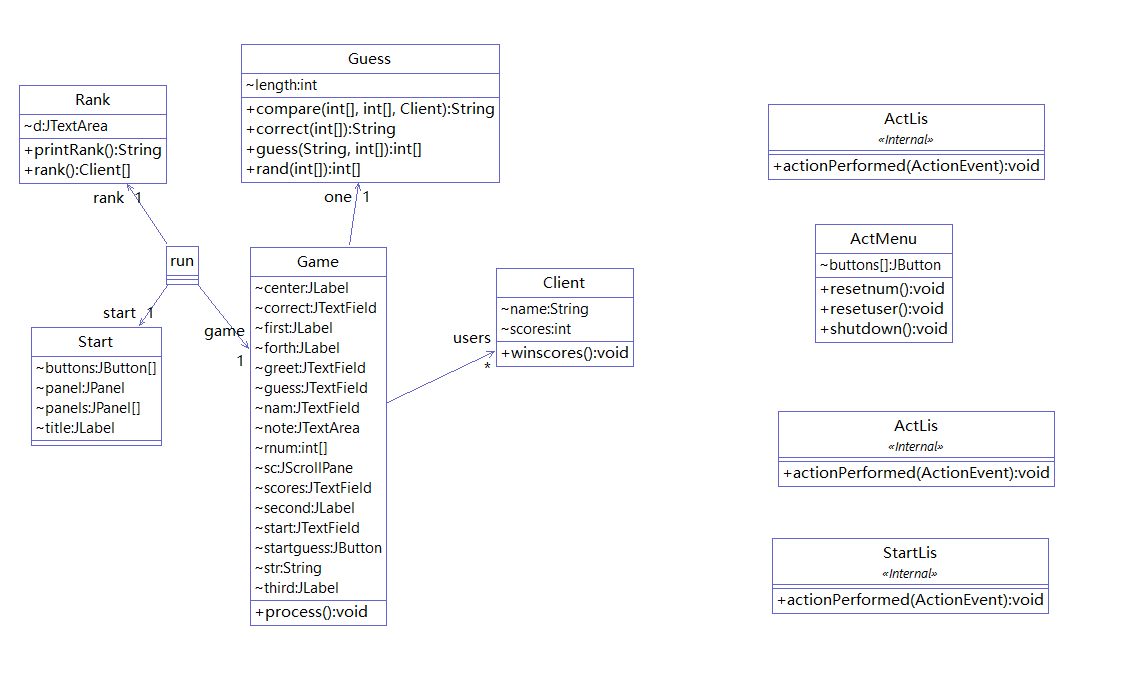
Client类 用户类 包含用户名和分数属性

Guess类 猜数字算法类

ActMenu类 含新用户 新数字 排行等功能

四．设计方案

1.UML类图



2.关键代码说明

a.根据用户猜的数组X和程序先前随机出的数组Y比较得出nAmB

具体实现：采用一个标记布尔数组exact，先将X与Y中的数字一对一比较，当数字与位置都相同时，令n自增，同时exact中取true，再将x中每一个数与y中每一个比较，略过那些数字位置都相同的，仅记录数字相同但位置不同的，令m自增，最终得出nAmB

**public** String compare(**int**[] rnum,**int**[] gnum,Client user){

**int** n=0,m=0,len=rnum.length;

**boolean**[] exact = **new** **boolean**[len];

**for**(**int** i=0;i<len;i++)

**if**(rnum[i]==gnum[i]){

n++;

exact[i]=**true**;

}

**for**(**int** i=0;i<len;i++){

**if**(exact[i])

**continue**;

**for**(**int** j=0;j<len;j++)

**if**(gnum[i]==rnum[j])

m++;

}

String result;

**if**(n==len){

result = n+"A"+" Correct!\n";

user.winscores();

}

**else**{

result = n+"A"+m+"B"+" Try again!\n";

}

**return** result;

}

b.开始猜数字的按钮监听器

按下按钮后实行以下操作：得到输入框里的用户名、猜测的数字、将该数字与原随机数进行比较得出nAmB并输出信息，同时更新分数

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == startguess) {

**int**[] gnum = **new** **int**[*length*];

String gstr=guess.getText();

gnum=one.guess(gstr,gnum);

str+=one.compare(rnum,gnum,*users*[*count*-1]);

note.setText(str);

*users*[*count*-1].name=nam.getText();

scores.setText("Scores of "+*users*[*count*-1].name+" :"+*users*[*count*-1].scores);

}

c.用选择排序给用户数组排名次

**public** Client[] rank(){

Client[] cli=**new** Client[Game.*count*];

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

cli[i]=Game.*users*[i];

**for**(**int** i=0;i<Game.*count*;i++){

Client temp=cli[i];

**int** n=i;

**for**(**int** j=i+1;j<Game.*count*;j++){

**if**(cli[j].scores>temp.scores){

temp=cli[j];

n=j;

}

}

cli[n]=cli[i];

cli[i]=temp;

}

**return** cli;

}

d.上方一横列功能按钮的监听器

按下“新用户”，在用户数组里创建一个用户对象，更新游戏版面信息（欢迎词，得分，随机数等），清空输入框内容，用户数+1

按下“猜新数”，随机新数字，清空输入框内容

按下“排名”，隐藏游戏界面，打开排名界面

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == buttons[0]) {

shutdown();

run.*frames*[1].setVisible(**true**);

Game.*count*++;

run.*game*.greet.setText(Client.*greetings*(Game.*count*));

resetuser();

Game.*users*[Game.*count*-1]=**new** Client();

run.*game*.process();

} **else** **if** (e.getSource() == buttons[1]) {

shutdown();

run.*frames*[1].setVisible(**true**);

resetnum();

run.*game*.process();

}

**else** **if** (e.getSource() == buttons[2]) {

shutdown();

run.*frames*[2].setVisible(**true**);

run.*rank*.d.setText(run.*rank*.printRank());

}

}