

广西师范大学漓江学院理工学院

课程综合设计报告

|  |  |
| --- | --- |
| **课程名称** | 数据结构 |
| **年级专业** | 2019级软件工程 |
| **选题名称** | 进制转换计算器 |
| **小组成员及**  **学号** | 201813005950 吴警 |
| **指导老师** | 杨嫘 |

**2021年7月6日**

目 录

[1 需求描述 3](#_Toc30567)

[2 技术选型 3](#_Toc25933)

[3 开发环境 3](#_Toc4519)

[4 人员分工 3](#_Toc16483)

[5 接口设计 4](#_Toc5890)

[6 源代码 5](#_Toc12318)

[7 项目运行 9](#_Toc32145)

[8 总结 11](#_Toc15314)

# 1 需求描述

实现输入一个初始数值，选择对应的初始进制和转换后进制然后输出一个目标进制数

# 2 技术选型

采用的技术：栈

特点：栈的优势是，存取速度比堆快，栈数据可以共享。但缺点是存放在栈中的数据占用多少内存空间需要在编译时确定下来，缺乏灵活性。

语言：Java，Java中有栈可直接调用（当然也可以自己写），可以做出合适窗体的窗体并编写容易使用的功能。

# 3 开发环境

硬件环境：MagicBook笔记本电脑 64 位操作系统

软件环境：Windows 10 20H2

开发工具：Eclipse IDE for Java Developers - 2021-06

# 4 人员分工

吴警：代码编写及注释；报告书撰写；查阅资料；代码测试与优化，GUI界面设计

# 5 接口设计

Stack类（自定义）：

定义栈容器、栈容量、栈顶位置和栈长度。

编写方法：

peek()方法获取栈顶元素。

isEmpty()方法判断栈是否为空。

isFull()方法判断栈是否已满。

push()方法实现入栈过程。

pop()方法实现出栈过程。

len()方法获取栈实际大小。

Decimalconversion类（）实现进制转换功能：

创建字符数值存储0-9及A-F

用Stack类的push()方法把符合规范的输入的数值入栈。

先将初始进制转为10进制。

逐位出栈，使用Arrays.binarySearch()方法 即二分查找查找栈中元素在字符数组中的位置，即此元素某一位的大小。比如21的“2”在字符数组中下标为2，表示这一位的大小为2。再乘以进制数的位数次方就可得到该位在10进制下的数值，累加即可得到10进制位数。

再将10进制数不断取余目标进制数入栈，最后返回最终所需进制数。

GUI类（图形界面）：

创建JTextField单行文本域用以输入数值和显示结果。

创建JRadioButton单选按钮组件和ButtonGroup按钮组用以构建进制选择机制，即原进制和最终进制只能选一个且不能相同（相同的话没有意义）。

创建addActionListener事件监听器，当点击按钮检查输入内容是否符合规范，不符合规范则跳出窗口提示，符合规范则调用Decimalconversion类的方法计算出结果并返回到指定文本框内。

Initsq：初始化顺序栈。

AllTurnAll：实现数的接收、处理以及输出转换后的数。

AllTurnAll里的参数：

num：对待转换的数进行接收。

b\_n：记录待转换数的进制。

g\_n：记录要转成的目标进制。

begin\_num：记录num的值，后面输出需要。

ten\_num：接收allTurnTen返回的十进制数。

result：接收tenTurnAll返回的目标进制数。

allTurnTen：利用Push和Pop将输入的数转化为十进制数并返回其值。

allTurnTen里的函数：

Push：对输入的数转换为十进制数压栈。

Push的参数：

num：将allTurnTen传入的数进行压栈。

Pop：将十进制数出栈并返回值给allTurnTen的t。

allTurnTen里的参数：

n：接收AllTurnAll里的b\_n的值。

num：接收AllTurnAll里的num的值。

t：记录num与10的余数然后接收Pop返回的值。

result：转换出来的十进制数。

tenTurnAll：利用Push和Pop将十进制数转换为目标进制数。

tenTurnAll里的参数：

num：接收AllTurnAll里的ten\_num的值。

n：接收AllTurnAll里的g\_n的值

Result：转换出来的目标进制数并返回。

t：记录num与n的余数然后接收Pop返回的值。

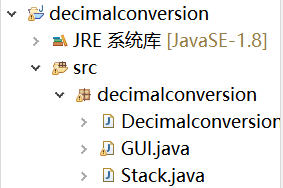
tenTurnAll里的Push：将十进制数转化为目标进制数进行压栈处理。

Push：的参数与上方Push参数一致。

tenTurnAll里的Pop：将目标进制数输出。

Pop的参数与上方Pop的参数一致。

# 6 源代码



GUI类：

package decimalconversion;//进制转换

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.ButtonGroup;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JRadioButton;

import javax.swing.JTextField;

public class GUI extends JFrame{

int ic,fc;

String regex = "";

public static void main(String[] args) {

new GUI();

}

public GUI() {

JFrame frame = new JFrame("进制转换");//创建一个JFrame窗体

frame.setLayout(null);//无布局

frame.setSize(600, 400);//设置窗体大小

frame.setLocationRelativeTo(null);//使窗体居中

JLabel l\_initialnumber = new JLabel("初始数值：");//创建一个JLabel标签

JLabel l\_initialconversion = new JLabel("初始进制：");

JLabel l\_finalconversion = new JLabel("转换后进制：");

JLabel l\_finalnumber = new JLabel("转换后数值：");

l\_initialnumber.setBounds(120, 30, 160, 30);//设置标签在窗体中的位置及标签大小

l\_initialconversion.setBounds(120, 90, 160, 30);

l\_finalconversion.setBounds(120, 150, 160, 30);

l\_finalnumber.setBounds(120, 210, 160, 30);

l\_initialnumber.setFont(new Font("宋体", Font.PLAIN, 25));//设置标签内字体格式

l\_initialconversion.setFont(new Font("宋体", Font.PLAIN, 25));

l\_finalconversion.setFont(new Font("宋体", Font.PLAIN,25));

l\_finalnumber.setFont(new Font("宋体", Font.PLAIN,25));

frame.add(l\_initialnumber);//添加组件进入窗体

frame.add(l\_initialconversion);

frame.add(l\_finalconversion);

frame.add(l\_finalnumber);

JTextField initialnumber =new JTextField(100);//创建一个JTextField单行文本域

JTextField finalnumber =new JTextField(100);

initialnumber.setEditable(true);//设置文本域内文字可写入

finalnumber.setEditable(false);//设置文本域内文字不可写入

initialnumber.setBounds(320, 30, 160, 30);//设置文本域在窗体中的位置及文本域大小

finalnumber.setBounds(320, 210, 160, 30);

initialnumber.setFont(new Font("宋体", Font.PLAIN, 25));//设置文本域内字体格式

finalnumber.setFont(new Font("宋体", Font.PLAIN,25));

frame.add(initialnumber);//添加组件进入窗体

frame.add(finalnumber);

JButton button1 = new JButton("运算");//创建一个JButton按钮

JButton button2 = new JButton("清空");

button1.setBounds(120, 270, 160, 30);//设置按钮在窗体中的位置及按钮大小

button2.setBounds(320, 270, 160, 30);

button1.setFont(new Font("宋体", Font.PLAIN, 20));//设置文本域内字体格式

button2.setFont(new Font("宋体", Font.PLAIN, 20));

frame.add(button1);//添加组件进入窗体

frame.add(button2);

//设置两套按钮组，分别代表初始进制和转换后进制，且两种进制设置为不相同

JPanel panel1=new JPanel(); //创建面板

JPanel panel2=new JPanel();

JRadioButton i2=new JRadioButton("2"); //创建JRadioButton对象

JRadioButton i8=new JRadioButton("8"); //创建JRadioButton对象

JRadioButton i10=new JRadioButton("10"); //创建JRadioButton对象

JRadioButton i16=new JRadioButton("16"); //创建JRadioButton对象

JRadioButton f2=new JRadioButton("2"); //创建JRadioButton对象

JRadioButton f8=new JRadioButton("8"); //创建JRadioButton对象

JRadioButton f10=new JRadioButton("10"); //创建JRadioButton对象

JRadioButton f16=new JRadioButton("16"); //创建JRadioButton对象

ButtonGroup group1=new ButtonGroup();

ButtonGroup group2=new ButtonGroup();

panel1.setBounds(320, 90, 160, 30);

panel2.setBounds(320, 150, 160, 30);

//添加JRadioButton到ButtonGroup中

group1.add(i2);

group1.add(i8);

group1.add(i10);

group1.add(i16);

group2.add(f2);

group2.add(f8);

group2.add(f10);

group2.add(f16);

panel1.add(i2);

panel1.add(i8);

panel1.add(i10);

panel1.add(i16);

panel2.add(f2);

panel2.add(f8);

panel2.add(f10);

panel2.add(f16);

frame.add(panel1);

frame.add(panel2);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);////用户单击窗口的关闭按钮时程序执行的操作

i2.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

i2.setVisible(true);

ic=2;

regex = "^[0-1] +$";//提前设置进制所对应的正则表达式

f2.setEnabled(false);//隐藏相同的进制数

f8.setEnabled(true);

f10.setEnabled(true);

f16.setEnabled(true);

}

});

i8.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

i8.setVisible(true);

ic=8;

regex = "^[0-7]+$";

f2.setEnabled(true);

f8.setEnabled(false);

f10.setEnabled(true);

f16.setEnabled(true);

}

});

i10.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

i10.setVisible(true);

ic=10;

regex = "^[0-9]+$";

f2.setEnabled(true);

f8.setEnabled(true);

f10.setEnabled(false);

f16.setEnabled(true);

}

});

i16.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

i16.setVisible(true);

ic=16;

regex = "^[A-F0-9]+$";

f2.setEnabled(true);

f8.setEnabled(true);

f10.setEnabled(true);

f16.setEnabled(false);

}

});

f2.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

f2.setVisible(true);

fc=2;

i2.setEnabled(false);

i8.setEnabled(true);

i10.setEnabled(true);

i16.setEnabled(true);

}

});

f8.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

f8.setVisible(true);

fc=8;

i2.setEnabled(true);

i8.setEnabled(false);

i10.setEnabled(true);

i16.setEnabled(true);

}

});

f10.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

f10.setVisible(true);

fc=10;

i2.setEnabled(true);

i8.setEnabled(true);

i10.setEnabled(false);

i16.setEnabled(true);

}

});

f16.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

f16.setVisible(true);

fc=16;

i2.setEnabled(true);

i8.setEnabled(true);

i10.setEnabled(true);

i16.setEnabled(false);

}

});

button1.addActionListener(new ActionListener() {//（运算按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

button1.setVisible(true);

String in;//定义初始数值，因为16进制数有字母，所以不能用int类型

try {

in = initialnumber.getText().trim();//获取文本域中内容赋值给in

if(in.matches(regex)) {

JOptionPane.showMessageDialog(frame, "填写的初始数值与初始进制不符合，请修改您的初始数值", "错误", JOptionPane.ERROR\_MESSAGE);//出现错误则显示对话框提示

}else {

finalnumber.setText(Decimalconversion.baseConvert(in, ic, fc));//调用TestStack类中baseConvert函数计算出最终结果并设置文本让其显示在文本域中

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(frame, "请检查输入是否有误", "错误", JOptionPane.ERROR\_MESSAGE);//抛出异常则显示对话框提示

}

}

});

button2.addActionListener(new ActionListener() {//（清空按钮）事件监听器

public void actionPerformed(ActionEvent e) {//（按钮）点击事件

frame.setVisible(true);

button1.setVisible(true);

try {

initialnumber.setText("");//将文本域中内容清空

finalnumber.setText("");

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(frame, "出现错误", "错误", JOptionPane.ERROR\_MESSAGE);

}

}

});

}

}

Stack类：

package decimalconversion;

public class Stack {

Object[] stacks; // 栈容器

int size; // 栈容量

int top; // 记录位置

int len; // 栈实际成员数量

public Stack(int size) {

super();

this.size = size;

this.stacks = new Object[this.size];

this.top = -1;

}

// 获取栈顶元素

public Object peek() {

return this.stacks[top];

}

// 判断栈是否为空

public boolean isEmpty() {

return top == (-1);

}

// 判断栈是否已满

public boolean isFull() {

return top == (size - 1);

}

// 入栈过程

public void push(Object value) {

len++;

stacks[++this.top] = value;

}

// 出栈过程

public Object pop() {

len--;

return stacks[this.top--]; // 注意这里不能用“--this.top”，会导致栈溢出

}

// 获取栈实际大小

public int len() {

return this.len;

}

}

Decimalconversion类：

package decimalconversion;

import java.util.Arrays;

public class Decimalconversion {

// 通用进制转换方法

public static String baseConvert(String initialnumber, int initialconversion,int finalconversion) {

Stack stack = new Stack(100);

char[] a=new char[]{'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};

char[] b=initialnumber.toCharArray();

int bNumber=0,t=0;

for (char p:b){

stack.push(p);

}

//将初始进制转为10进制

while (!stack.isEmpty()){

char temp=(char) stack.pop();

bNumber+= Arrays.binarySearch(a,temp)\*Math.pow(initialconversion,t);

t++;

}

//将10进制转为最终所需进制

StringBuilder builder=new StringBuilder();

while (bNumber>0){

stack.push(a[bNumber%finalconversion]);

bNumber/=finalconversion;

}

while (!stack.isEmpty()) {

builder.append(stack.pop());

}

String str;

str=builder.toString();

return str;

}

}

# 7 项目运行

初始界面：



二进制转换为其他进制：





八进制转换为其他进制：





十进制转换为其他进制：



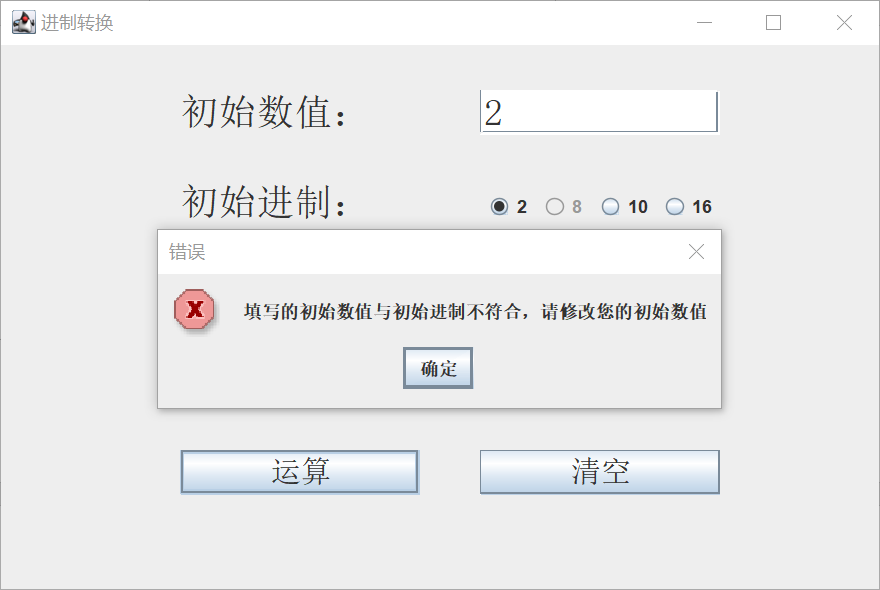


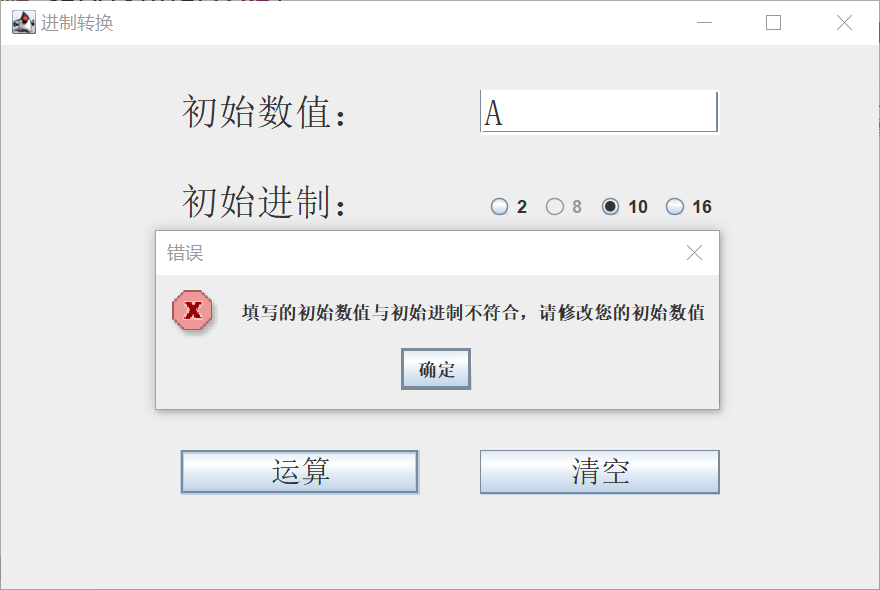
十六进制转换为其他进制：



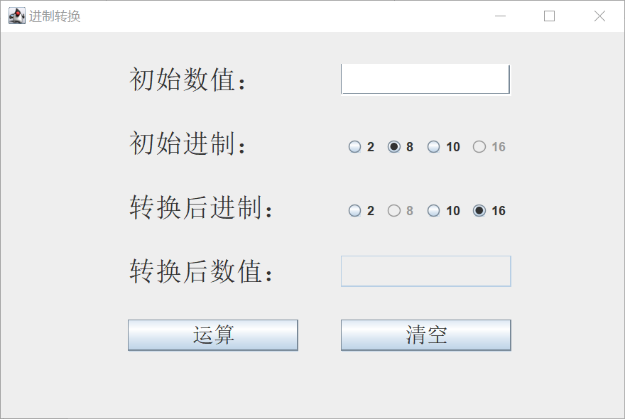


填写不正确：





清空效果：



# 8 总结

一：在处理图形化界面和数值判断花了非常长的时间，尤其是后者，对一些问题的处理仍不够熟练。

二：这个程序运用了堆栈的相关知识，编程不难，但是也让我们发现自己在编程上的许多问题，以及对Java语言的方法调用处理和语法还不够熟悉，但知道了不足，就更加明确需要加强的地方是什么。