作业1

自己写一个简单Java程序，包含四则运算和if for语句，分析对应的字节码

编写Java测试代码如下

package Week1.homework;  
  
public class TestByteCode {  
 public static void main(String[] args) {  
 int a = 100;  
 int b = 33;  
  
 int add\_val = a + b;  
 int sub\_val = a - b;  
 int mul\_val = a \* b;  
 int div\_val = a / b;  
  
 if (a <= 100) {  
 a = 10;  
 }  
  
 for (int i = 0; i < a; i++) {  
 b += 1;  
 }  
 }  
}

用javap工具生成的字节码文件中核心的局部变量表信息如下：

*LocalVariableTable:  
 Start Length Slot Name Signature  
 37 15 7 i I  
 0 53 0 args [Ljava/lang/String;  
 3 50 1 a I  
 6 47 2 b I  
 10 43 3 add\_val I  
 15 38 4 sub\_val I  
 20 33 5 mul\_val I  
 25 28 6 div\_val I*

其中0号槽位被静态函数main的参数args占据，1~6五个槽位分别对应main函数中内部的6个局部变量

核心的main函数对应的字节码如下，对其中每一条指令的解析入注释所示

*/\*  
字节码  
  
public static void main(java.lang.String[]);  
 descriptor: ([Ljava/lang/String;)V  
 flags: (0x0009) ACC\_PUBLIC, ACC\_STATIC  
 Code:  
 stack=2, locals=8, args\_size=1 // 函数栈分配2个32bit空间，局部变量表8个槽位  
 0: bipush 100 // 100常量放入栈顶  
 2: istore\_1 // 栈顶的常量100退栈，放入1 槽位，即赋值a=100  
 3: bipush 33 // 33常量放入栈顶  
 5: istore\_2 // 栈顶的常量33退栈，放入2槽位，即赋值b=33  
 6: iload\_1 // a变量放入栈顶  
 7: iload\_2 // b 变量放入栈顶  
 8: iadd // 进行a+b计算，结果保留在栈顶  
 9: istore\_3 // 栈顶a+b的结果弹栈放入3槽位，即赋值add\_val = a+b  
 10: iload\_1 // a 变量放入栈顶  
 11: iload\_2 // b变量放入栈顶  
 12: isub // 执行a-b计算，结果放到栈顶  
 13: istore 4 // 栈顶a-b结果弹栈放入4槽位，即赋值sub\_val = a-b  
 15: iload\_1 // a变量放入栈顶  
 16: iload\_2 // b变量放入栈顶  
 17: imul // 执行a\*b计算，结果放入栈顶  
 18: istore 5 // 栈顶a\*b计算结果弹栈放入槽位5，即赋值mul\_val = a\*b  
 20: iload\_1 // a变量放入栈顶  
 21: iload\_2 // b变量放入栈顶  
 22: idiv // 执行a/b计算，结果放入栈顶  
 23: istore 6 // 栈顶a/b结果弹栈放入6槽位，即赋值div\_val = a/b  
 25: iload\_1 // a变量放入栈顶  
 26: bipush 100 // 常量100放入栈顶  
 28: if\_icmpgt 34 // 比较a > 100 是否成立，如果成立跳转34位置执行  
 31: bipush 10 // 常量10放入栈顶  
 33: istore\_1 // 此时a<=10成立，执行a = 10语句  
 34: iconst\_0 // 常量0放入栈顶  
 35: istore 7 // 栈顶0数值弹栈放入7槽位, 即i= 0  
 37: iload 7 // 加载i变量数值到栈顶  
 39: iload\_1 // 加载a到栈顶  
 40: if\_icmpge 52 // 比较i >= a是否成立，如果成立跳转52位置执行，循环结束  
 43: iinc 2, 1 // b+=1语句执行  
 46: iinc 7, 1 // i++ 语句执行  
 49: goto 37 // 循环回到37位置执行  
 52: return  
  
  
 \*/*