enum AllOperator｛｝

enum TypeROperator｛｝

enum TypeIOperator｛｝

enum TypeJOperator｛｝

若enum无法实现或者较难实现遍历，则用数组代替，并定义三个final变量：NumOfR，NumOfI，NumOfJ

Enum.values应该可以实现遍历

Assembler

field：TypeRDecoder typeRDecoder = new

TypeIDecoder typeIDecoder = new

TypeJDecoder typeJDecoder = new

String[] instruction

number numOfIns

number[][] binCode

methods： constructor（）

void setInstruction(String)

boolean checkChars（）

boolean checkOperator（）

Char getType（）

TypeRDecoder

field：String instruction

String operator

String rs

String rt

String rd

final number numOfOperand = 3

method：

网页加载后创建assembler

Assembler assembler = new Assembler（）

点击load按钮后调用loadInstruction

Onclick = loadInstruction（document.getElementById（ins）.value）

void loadInstruction（String）：

assembler.setInstruction(String ins)

setInstruction根据\n的情况将指令存入assemble对象的instruction字符串数组并且判断指令个数，设置numOfIns. <texarea>支持多行输入.该函数需进行以下判断：1.若第一个字符就是换行符，跳过换行符读取下一个字符，若同样是换行符，继续跳过，直到不是换行符为止。2.如果连续出现了多个换行符，则跳过这些换行符读取后面的内容。3.判断出共有几条指令，并set numOfIns,初始化instruction = new String[numOfIns], binCode = new number[numOfIns][32] 4.初始化i = 0.每次读取一条指令后将指令存入对应的instruction[i],并且i++。

for(i=0;i<numOfIns;i++){进行以下操作}

checkChars 检查输入(instruction[i])中是否为字母、数字、逗号、空格，最多三个$。正确则进行下一步，错则报错。if（checkChars）｛下一步｝else｛报错并且return｝

checkOperator截取第一个逗号前的字符串，遍历AllOperator，检查是否为正确的MIPS操作符。if（checkOperator）｛下一步｝else｛报错并且return｝

getType 截取第一个逗号前的字符串，遍历三种enum，检查是否符合其中任意一个，若符合则返回字符R、I或者J

根据getType的返回值进行判断（switch），并将instruction传入TypeXDecoder。

TypeXDecoder

Debug部分：

1.根据$的个数检测operand的数量是否符合该类型

2.根据各个operand可取值的范围进行判断是否有效

若说传入的指令为有效指令则根据操作符和操作数生成binary code，并传入binCode[i][]