1. ADDinstr:

EXPRSTACK[--ESP-1] += EXPRSTACK[ESP];

1. ANDinstr extends ZeroOperandInstruction:

EXPRSTACK[--ESP - 1] = EXPRSTACK[ESP - 1] & EXPRSTACK[ESP];

1. CALLSTATMETHODinstr extends OneOperandInstruction:

TJ.data[(ASP - POINTERTAG)] = PC;

ASP++;

PC = this.operand;

1. CHANGESIGNinstr extends ZeroOperandInstruction:

EXPRSTACK[ESP - 1] \*= -1;

1. DIVinstr extends ZeroOperandInstruction:

EXPRSTACK[--ESP - 1] = EXPRSTACK[ESP - 1] / EXPRSTACK[ESP];

1. EQinstr extends ZeroOperandInstruction:

if(EXPRSTACK[--ESP - 1] == EXPRSTACK[ESP]) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP - 1] = 0;

}

1. GEinstr extends ZeroOperandInstruction:

if(EXPRSTACK[--ESP - 1] >= EXPRSTACK[ESP]) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP - 1] = 0;

}

1. GTinstr extends ZeroOperandInstruction:

if(EXPRSTACK[--ESP - 1] > EXPRSTACK[ESP]) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP-1] = 0;

}

1. INITSTKFRMinstr extends OneOperandInstruction:

TJ.data[ASP - POINTERTAG] = FP;

FP = ASP++;

ASP = ASP + this.operand;

1. JUMPinstr extends OneOperandInstruction:

PC = this.operand;

1. JUMPONFALSEinstr extends OneOperandInstruction:

if(EXPRSTACK[--ESP] == 0) {

PC = this.operand;

}

1. LEinstr extends ZeroOperandInstruction:

if(EXPRSTACK[--ESP - 1] <= EXPRSTACK[ESP]) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP - 1] = 0;

}

1. LOADFROMADDRinstr extends ZeroOperandInstruction:

int a = EXPRSTACK[ESP - 1];

EXPRSTACK[ESP - 1] = TJ.data[a - POINTERTAG];

1. LTinstr extends ZeroOperandInstruction:

if(EXPRSTACK[--ESP - 1] < EXPRSTACK[ESP]) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP - 1] = 0;

}

1. MODinstr extends ZeroOperandInstruction:

EXPRSTACK[--ESP - 1] = EXPRSTACK[ESP - 1] % EXPRSTACK[ESP];

1. MULinstr extends ZeroOperandInstruction:

EXPRSTACK[--ESP - 1] = EXPRSTACK[ESP - 1] \* EXPRSTACK[ESP];

1. NEinstr extends ZeroOperandInstruction:

if(EXPRSTACK[--ESP - 1] != EXPRSTACK[ESP]) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP - 1] = 0;

}

1. NOTinstr extends ZeroOperandInstruction:

if(EXPRSTACK[ESP - 1] == 0) {

EXPRSTACK[ESP - 1] = 1;

} else {

EXPRSTACK[ESP - 1] = 0;

}

1. ORinstr extends ZeroOperandInstruction:

EXPRSTACK[--ESP - 1] = EXPRSTACK[ESP - 1] | EXPRSTACK[ESP];

1. PASSPARAMinstr extends ZeroOperandInstruction:

TJ.data[ASP++ - POINTERTAG] = EXPRSTACK[--ESP];

1. PUSHLOCADDRinstr extends OneOperandInstruction:

EXPRSTACK[ESP++] = FP + this.operand;

1. PUSHNUMinstr extends OneOperandInstruction:

EXPRSTACK[ESP++] = this.operand;

1. PUSHSTATADDRinstr extends OneOperandInstruction:

EXPRSTACK[ESP++] = this.operand + POINTERTAG;

1. RETURNinstr extends OneOperandInstruction:

ASP = FP;

FP = TJ.data[(ASP--) - POINTERTAG];

PC = TJ.data[ASP - POINTERTAG];

ASP -= this.operand;

1. SAVETOADDRinstr extends ZeroOperandInstruction:

int a = EXPRSTACK[ESP - 2] - POINTERTAG;

TJ.data[a] = EXPRSTACK[ESP - 1];

ESP = ESP - 2;

1. SUBinstr extends ZeroOperandInstruction:

EXPRSTACK[--ESP - 1] = EXPRSTACK[ESP - 1] - EXPRSTACK[ESP];

1. WRITEINTinstr extends ZeroOperandInstruction:

System.out.print(EXPRSTACK[--ESP]);

1. WRITELNOPinstr extends ZeroOperandInstruction:

System.out.println();

1. WRITESTRINGinstr extends TwoOperandInstruction:

for(int a = this.firstOperand; a <= this.secondOperand; a++) {

System.out.print((char)TJ.data[a]);

}