|  |
| --- |
| from  sys import  exit |
| motOpCode = { |
| "MOV": 1, |
| "A": 2, |
| "S": 3, |
| "M": 4, |
| "D": 5, |
| "AN": 6, |
| "O": 7, |
| "ADD": 8, |
| "SUB": 9, |
| "MUL": 10, |
| "DIV": 11, |
| "AND": 12, |
| "OR": 13, |
| "LOAD": 14, |
| "STORE": 15, |
| "DCR": 16, |
| "INC": 17, |
| "JMP": 18, |
| "JNZ": 19, |
| "HALT": 20 |
| } |
| motSize = { |
| "MOV": 1, |
| "A": 1, |
| "S": 1, |
| "M": 1, |
| "D": 1, |
| "AN": 1, |
| "O": 1, |
| "ADD": 1, |
| "SUB": 2, |
| "MUL": 2, |
| "DIV": 2, |

|  |
| --- |
| "AND": 2, |
| "OR ": 2, |
| "LOAD": 3, |
| "STORE": 3, |
| "DCR": 1, |
| "INC": 1, |
| "JMP": 3, |
| "JNZ": 3, |
| "HALT": 1 |
| } |
| l = [] |
| relativeAddress = [] |
| machineCode = [] |
| symbol = [] |
| symbolValue = [] |
| RA = 0 |
| current = 0 |
| count = 0 |
| temp = [] |
| n = int(input("Enter the no of instruction lines : ")) |
| for i in range(n): |
| instructions = input("Enter instruction line {} : ".format(i + 1)) |
| l.append(instructions) |
| l = [x.upper() for x in l] |
| for i in range(n): |
| x = l[i] |
| if "NEXT:" in x: |
| s1 = ''.join(x) |
| a, b, c = s1.split() |
| a = a[:4] |
| l[i] = b + " " + c |
| symbol.append(a) |
| x = l[i] |
| if b in motOpCode: |
| value = motOpCode.get(b) |
| size = motSize.get(b) |
| if len(str(size)) == 1: |
| temp = "000" + str(size) |
| elif len(str(size)) == 2: |
| temp = "00" + str(size) |
| elif len(str(size)) == 3: |
| temp = "0"+str(size) |
| else: |
| print("Instruction is not in Op Code.") |

|  |
| --- |
| exit(0) |
| symbolValue.append(temp) |
| previous = size |
| RA += current |
| current = previous |
| relativeAddress.append(RA) |
| if c.isalpha() is True: |
| machineCode.append(str(value)) |
| else: |
| temp = list(b) |
| for i in range(len(temp)): |
| if count == 2: |
| temp.insert(i, ',') |
| count = 0 |
| else: |
| count = count + 1 |
| s = ''.join(temp) |
| machineCode.append(str(value) + "," + s) |
| elif " " in x: |
| s1 = ''.join(x) |
| a, b = s1.split() |
| if a in motOpCode: |
| value = motOpCode.get(a) |
| size = motSize.get(a) |
| previous = size |
| RA += current |
| current = previous |
| relativeAddress.append(RA) |
| if b.isalpha() is True: |
| machineCode.append(str(value)) |
| else: |
| temp = list(b) |
| for i in range(len(temp)): |
| if count == 2: |
| temp.insert(i, ',') |
| count = 0 |
| else: |
| count = count + 1 |
| s = ''.join(temp) |
| machineCode.append(str(value) + "," + s) |
| else: |
| print("Instruction is not in Op Code.") |
| exit(0) |
| else: |
| if x in motOpCode: |
| value = motOpCode.get(x) |

|  |  |  |
| --- | --- | --- |
| size = motSize.get(x) | | |
| previous = size | | |
| RA += current | | |
| current = previous | | |
| relativeAddress.append(RA) | | |
| machineCode.append(value) | | |
| else: | | |
| print("Instruction is not in Op Code.") | | |
| exit(0) | | |
| print("Symbol Table : | \n") |  |
| print("\n Symbol | Value(Address)") |  |
| for i in range(len(symbol)): | | |
| print(" {} | {}".format(symbol[i], symbolValue[i])) |  |
| print("\n Pass-1 machine code output without reference of the symbolic address  : \n") | | |
| print("Relative Address | Instruction OpCode") |  |
| for i in range(n): | | |
| if "NEXT" in l[i]: | | |
| print("{}  ".format( | {} | {}, - |
| relativeAddress[i], l[i], machineCode[i])) | | |
| else: | | |
| print("{}  ".format( | {} | {} |
| relativeAddress[i], l[i], machineCode[i])) | | |
| print("\n Pass-2 output: Machine code output \n ") | | |
| print("Relative Address | Instruction OpCode") |  |
| for i in range(n): | | |
| if "NEXT" in l[i]: | | |
| for j in range(len(symbol)): | | |
| if "NEXT" in symbol[j]: | | |
| pos = j | | |
| print("{} {}  {} , {}".format( | | |
| relativeAddress[i], l[i], machineCode[i],  symbolValue[pos])) | | |
| else: | | |
| print("{}  ".format( | {} | {} |
| relativeAddress[i], l[i], machineCode[i])) | | |

