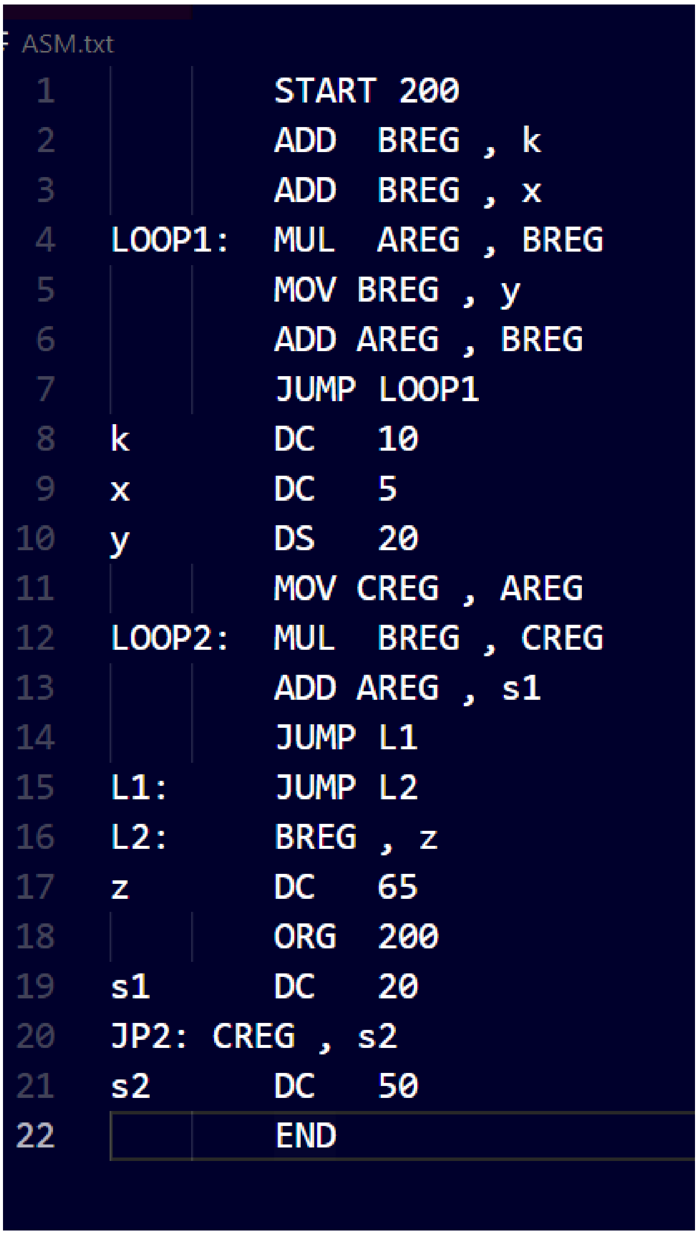
ASSIGNMENT 1
SYSTEM PROGRAMMING LAB

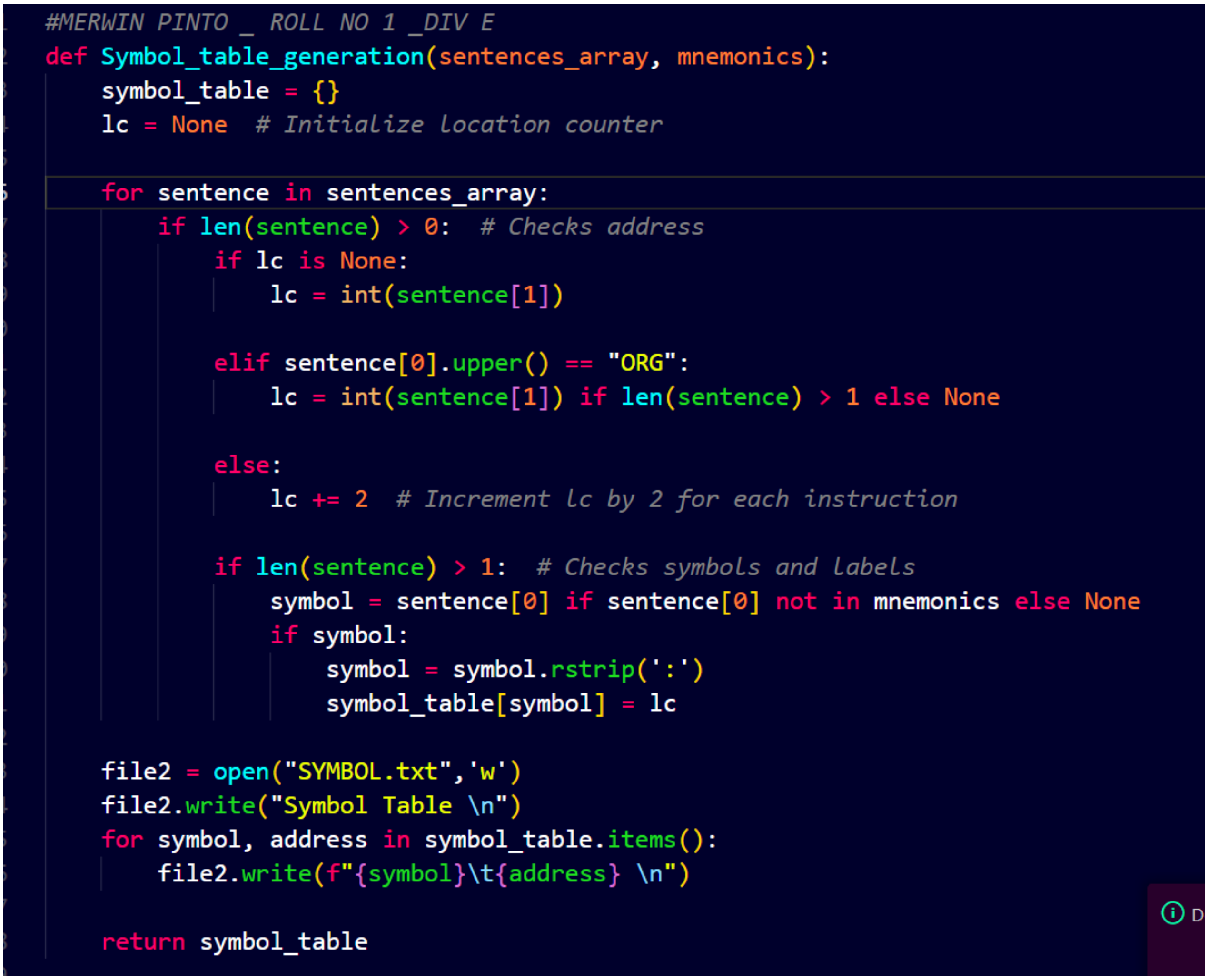
202100102
MERWIN PINTO
ROLL NO 1
DIV E

ASM FILE



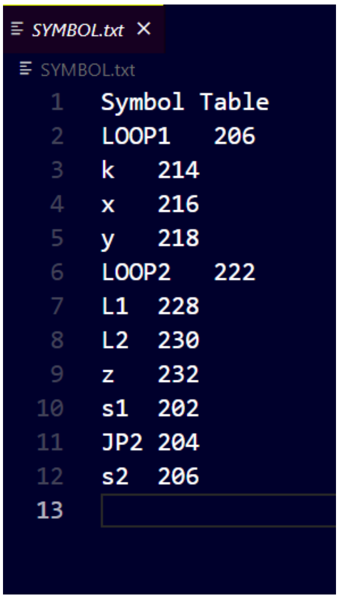
```
        START  200
        ADD  BREG , k
        ADD  BREG , x
LOOP1:  MUL  AREG , BREG
        MOV BREG , y
        ADD AREG , BREG
        JUMP LOOP1
k       DC   10
x       DC   5
y       DS   20
        MOV CREG , AREG
LOOP2:  MUL  BREG , CREG
        ADD AREG , s1
        JUMP L1
L1:     JUMP L2
L2:     BREG , z
z       DC   65
        ORG  200
s1      DC   20
JP2: CREG , s2
s2      DC   50
        END
```

SYMBOL TABLE GENERATE



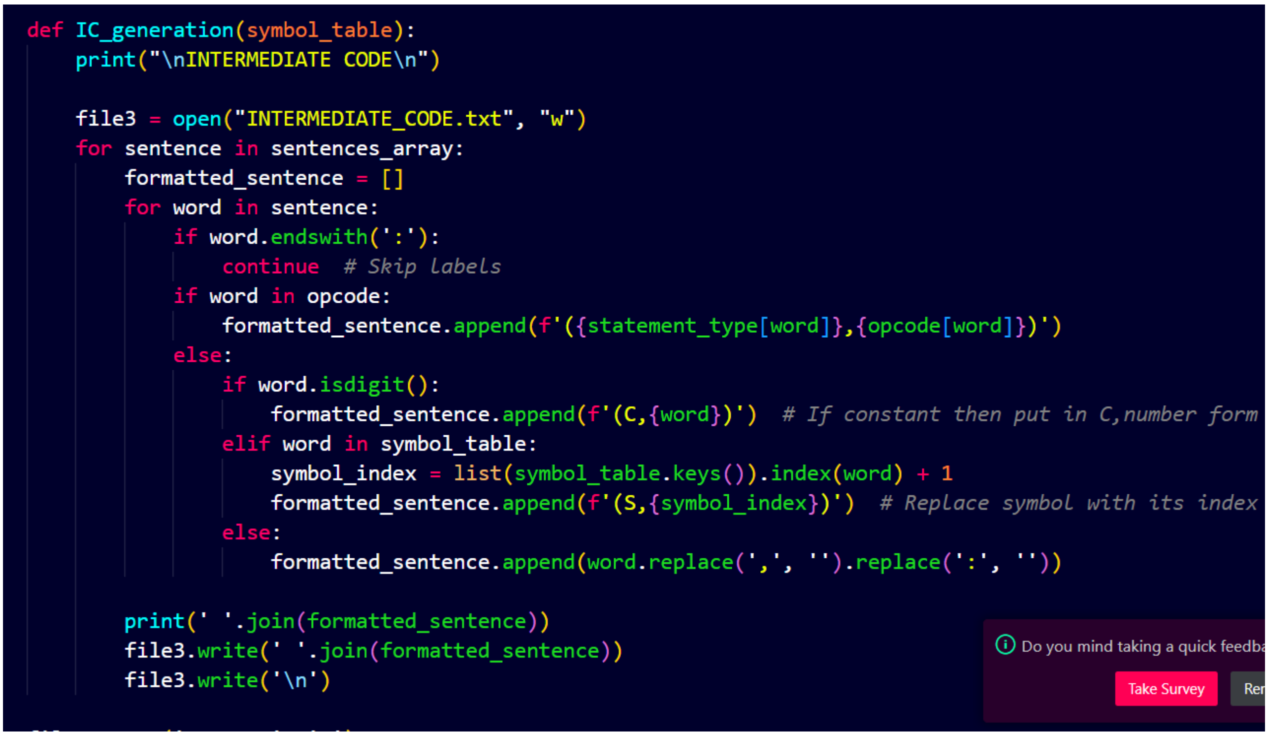
```
#MERWIN PINTO _ ROLL NO 1 _DIV E
def Symbol_table_generation(sentences_array, mnemonics):
    symbol_table = {}
    lc = None  # Initialize location counter

    for sentence in sentences_array:
        if len(sentence) > 0:  # Checks address
            if lc is None:
                lc = int(sentence[1])

            elif sentence[0].upper() == "ORG":
                lc = int(sentence[1]) if len(sentence) > 1 else None

            else:
                lc += 2  # Increment lc by 2 for each instruction

            if len(sentence) > 1:  # Checks symbols and labels
                symbol = sentence[0] if sentence[0] not in mnemonics else None
                if symbol:
                    symbol = symbol.rstrip(':')
                    symbol_table[symbol] = lc

    file2 = open("SYMBOL.txt",'w')
    file2.write("Symbol Table \n")
    for symbol, address in symbol_table.items():
        file2.write(f"{symbol}\t{address} \n")

    return symbol_table
```

OUTPUT:



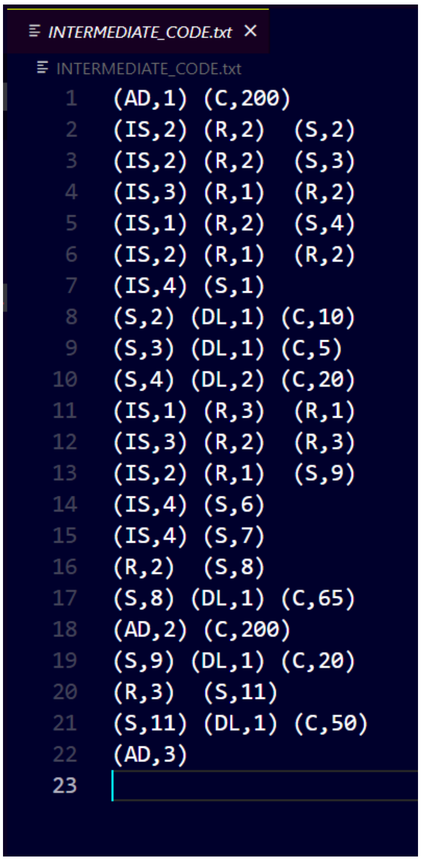
```
Symbol Table
LOOP1    206
k    214
x    216
y    218
LOOP2    222
L1   228
L2   230
z    232
s1   202
JP2  204
s2   206
```

INTERMEDIATE CODE GENERATION



```
def IC_generation(symbol_table):
    print("\nINTERMEDIATE CODE\n")

    file3 = open("INTERMEDIATE_CODE.txt", "w")
    for sentence in sentences_array:
        formatted_sentence = []
        for word in sentence:
            if word.endswith(':'):
                continue  # Skip labels
            if word in opcode:
                formatted_sentence.append(f'({statement_type[word]},{opcode[word]})')
            else:
                if word.isdigit():
                    formatted_sentence.append(f'(C,{word})')  # If constant then put in C,number form
                elif word in symbol_table:
                    symbol_index = list(symbol_table.keys()).index(word) + 1
                    formatted_sentence.append(f'(S,{symbol_index})')  # Replace symbol with its index
                else:
                    formatted_sentence.append(word.replace(',', '').replace(':', ''))

        print(' '.join(formatted_sentence))
        file3.write(' '.join(formatted_sentence))
        file3.write('\n')
```

OUTPUT



```
(AD,1) (C,200)
(IS,2) (R,2)  (S,2)
(IS,2) (R,2)  (S,3)
(IS,3) (R,1)  (R,2)
(IS,1) (R,2)  (S,4)
(IS,2) (R,1)  (R,2)
(IS,4) (S,1)
(S,2) (DL,1) (C,10)
(S,3) (DL,1) (C,5)
(S,4) (DL,2) (C,20)
(IS,1) (R,3)  (R,1)
(IS,3) (R,2)  (R,3)
(IS,2) (R,1)  (S,9)
(IS,4) (S,6)
(IS,4) (S,7)
(R,2)  (S,8)
(S,8) (DL,1) (C,65)
(AD,2) (C,200)
(S,9) (DL,1) (C,20)
(R,3)  (S,11)
(S,11) (DL,1) (C,50)
(AD,3)
```