Ishita Hardasmalani

C14-2103058

EXPERIMENT NO.7

Aim : Write a program to implement Pass 1 of Multi-Pass Assembler

Write a program to implement Pass 2 of Multi-Pass Assembler

Code:

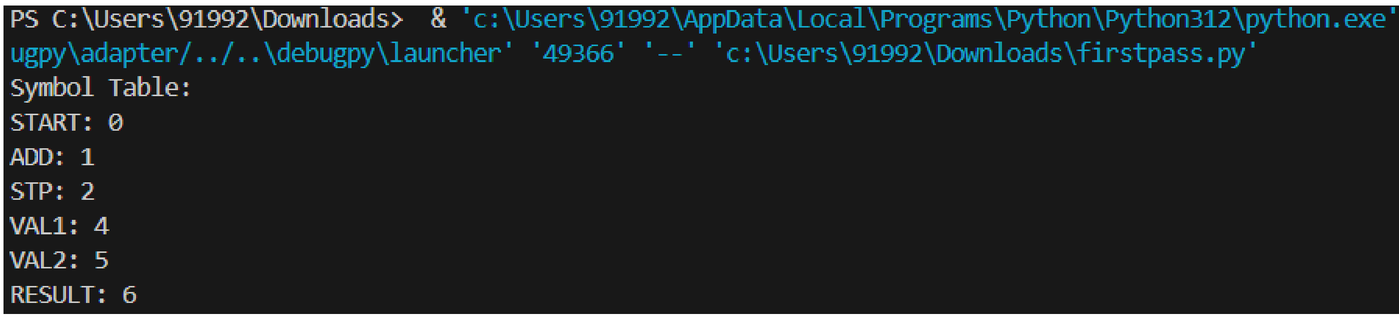
Source_Code.asm

```
START    LDA     VAL1      ; Load accumulator with value
        ADD     VAL2     ; Add value 2 to accumulator
        STP     RESULT    ; Store the result in memory
        HLT             ; Halt the program
VAL1     DAT      1      ; Data: value 1
VAL2     DAT      2      ; Data: value 2
RESULT   DAT      0      ; Data: Result
```

firstpass.py

```
def first_pass_assembler(source_code):
    symbol_table = {}
    location_counter = 0

    with open(source_code, 'r') as file:
        for line in file:
            line = line.split(';')[0].strip()
            if not line:
                continue

            tokens = line.split()
            label = None
        if len(tokens) > 1:
            label = tokens[0]

        if label and label in symbol_table:
            print(f"Error: Duplicate label '{label}'")
            return None

        if label:
            symbol_table[label] = location_counter

        location_counter += 1

    return symbol_table

# Example usage:
source_code = "Source_Code.asm"
symbol_table = first_pass_assembler(source_code)
if symbol_table:
    print("Symbol Table:")
    for label, location in symbol_table.items():
        print(f"{label}: {location}")
```

Output:



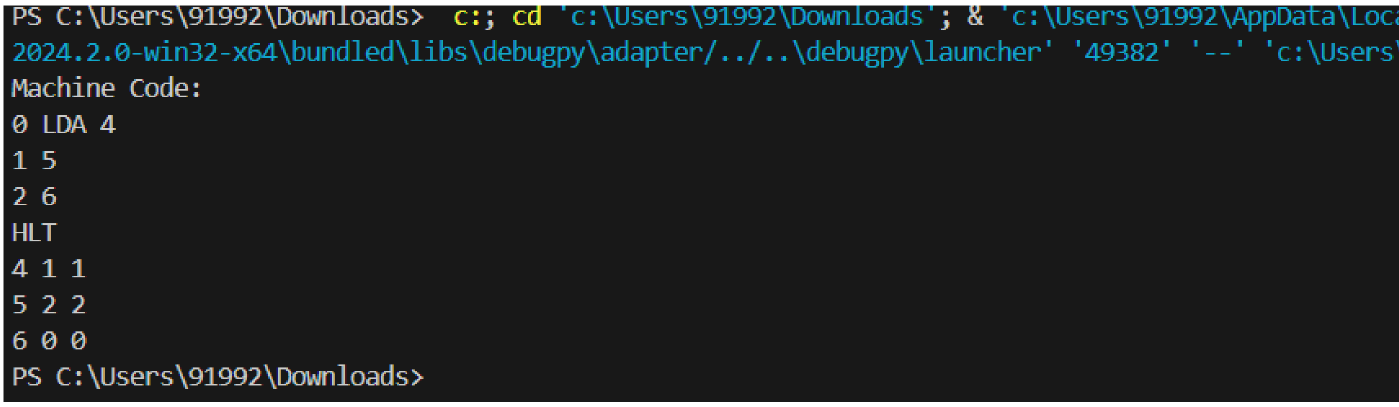
```
PS C:\Users\91992\Downloads>  & 'c:\Users\91992\AppData\Local\Programs\Python\Python312\python.exe'
ugpy\adapter/../..\debugpy\launcher' '49366' '--' 'c:\Users\91992\Downloads\firstpass.py'
Symbol Table:
START: 0
ADD: 1
STP: 2
VAL1: 4
VAL2: 5
RESULT: 6
```

secondpass.py

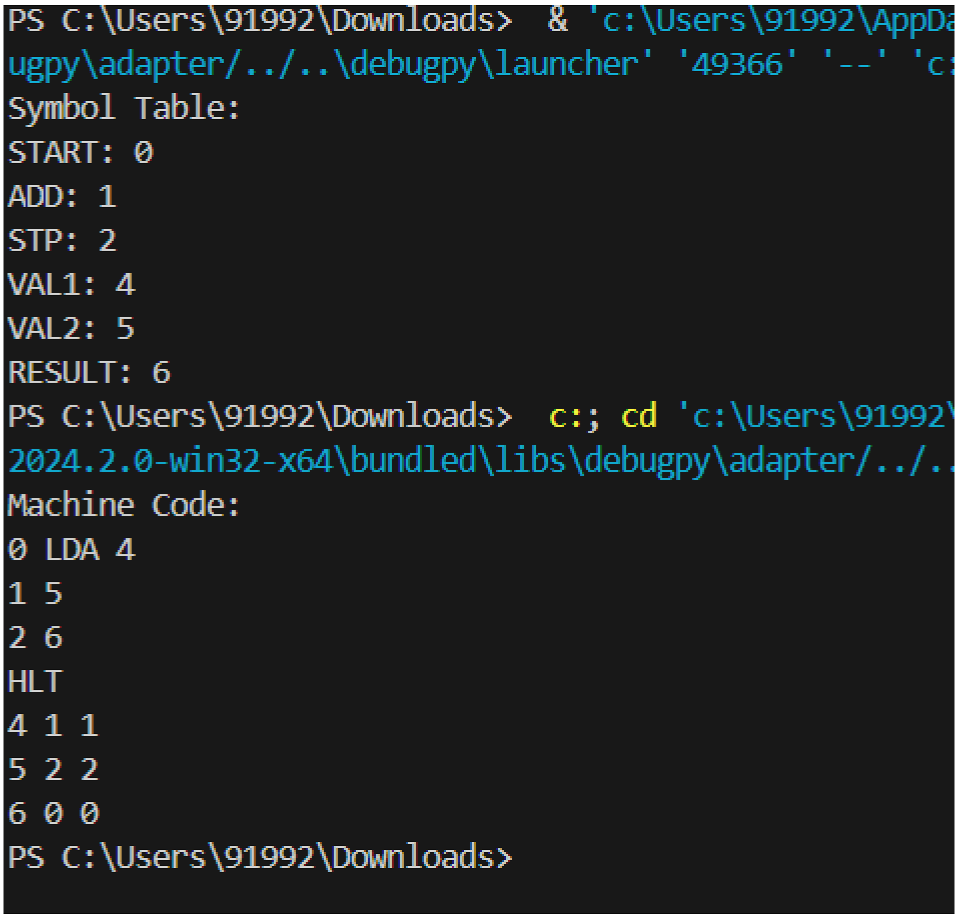
```
def pass2_assembler(source_code, symbol_table):
    machine_code = []

    with open(source_code, 'r') as file:
        for line in file:
            line = line.split(';')[0].strip()
            if not line:
                continue

            tokens = line.split()
            translated_instruction = ''

            for i, token in enumerate(tokens):
                if token.isdigit():
                    translated_instruction += token + ' '
                elif token in symbol_table:
                    translated_instruction += str(symbol_table[token]) + ' '
                elif token == 'DAT':
                    translated_instruction += tokens[i+1] + ' '
                else:
                    translated_instruction += token + ' '

            machine_code.append(translated_instruction.strip())

    return machine_code

# Example usage:
source_code = "Source_Code.asm"
```

```
symbol_table = {
    'START': 0,
    'ADD': 1,
    'STP': 2,
    'VAL1': 4,
    'VAL2': 5,
    'RESULT': 6
}
machine_code = pass2_assembler(source_code, symbol_table)
print("Machine Code:")
for instruction in machine_code:
    print(instruction)
```

Output:



```
PS C:\Users\91992\Downloads>  c:; cd 'c:\Users\91992\Downloads'; & 'c:\Users\91992\AppData\Loca
2024.2.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '49382' '--' 'c:\Users\
Machine Code:
0 LDA 4
1 5
2 6
HLT
4 1 1
5 2 2
6 0 0
PS C:\Users\91992\Downloads>
```



```
PS C:\Users\91992\Downloads>  & 'c:\Users\91992\AppDa
ugpy\adapter/../..\debugpy\launcher' '49366' '--' 'c:
Symbol Table:
START: 0
ADD: 1
STP: 2
VAL1: 4
VAL2: 5
RESULT: 6
PS C:\Users\91992\Downloads>  c:; cd 'c:\Users\91992\
2024.2.0-win32-x64\bundled\libs\debugpy\adapter/../..
Machine Code:
0 LDA 4
1 5
2 6
HLT
4 1 1
5 2 2
6 0 0
PS C:\Users\91992\Downloads>
```