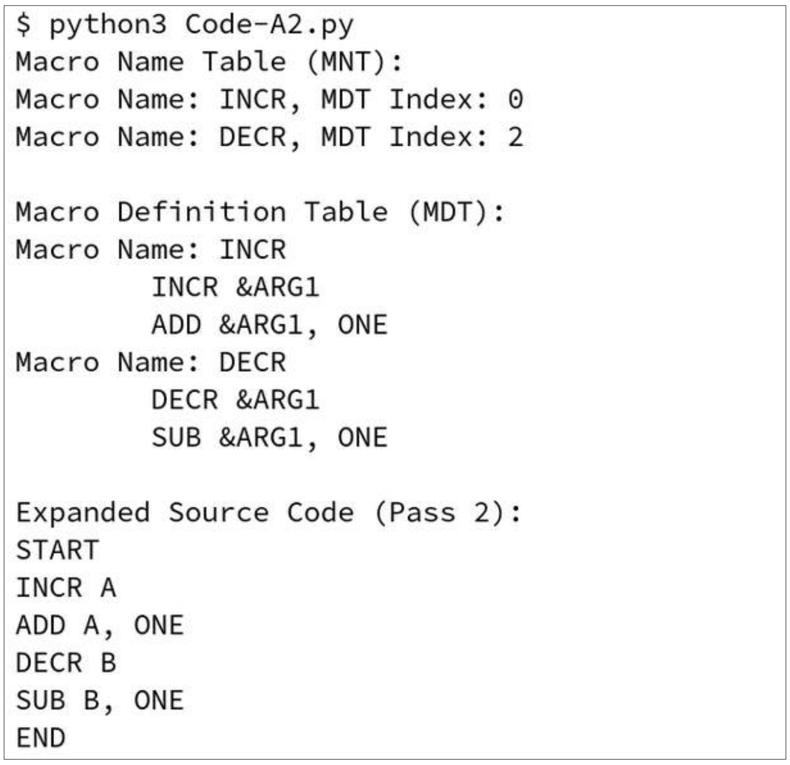
Assignment – A2

Code

```
 class MacroProcessor:
     def __init__(self):
         self.macro_name_table = {}
         self.macro_definition_table = {}
         self.mdt_index = 0

     def process_pass1(self, source_code):
         inside_macro = False
         current_macro_name = None
         macro_definition = []

         for line in source_code:
             tokens = line.strip().split()

             if not tokens:
                 continue

             if tokens[0] == 'MACRO':
                 inside_macro = True
                 continue

             if inside_macro and tokens[0] == 'MEND':
                 inside_macro = False
                 self.macro_definition_table[current_macro_name] =
macro_definition[:]
                 self.macro_name_table[current_macro_name] =
self.mdt_index
                 self.mdt_index += len(macro_definition)
                 macro_definition = []
                 current_macro_name = None
                 continue

             if inside_macro:
                 if not current_macro_name:
                     current_macro_name = tokens[0]
                 macro_definition.append(line.strip())
             else:
                 pass

    def process_pass2(self, source_code):
         output = []
         inside_macro = False

         for line in source_code:
             tokens = line.strip().split()

             if not tokens:
                 continue
```

```
            if tokens[0] == 'MACRO':
                inside_macro = True
                continue
            elif tokens[0] == 'MEND':
                inside_macro = False
                continue

            if inside_macro:
                continue

            macro_name = tokens[0]
            if macro_name in self.macro_name_table:
                macro_def = self.macro_definition_table[macro_name]
                args = tokens[1:]
                output.extend(self.expand_macro(macro_def, args))
            else:
                output.append(line.strip())

        return output

    def expand_macro(self, macro_def, args):
        expanded_lines = []
        for line in macro_def:
            for i, arg in enumerate(args):
                line = line.replace(f"&ARG{i+1}", arg)
            expanded_lines.append(line)
        return expanded_lines

    def display_tables(self):
        print("Macro Name Table (MNT):")
        for name, index in self.macro_name_table.items():
            print(f"Macro Name: {name}, MDT Index: {index}")

        print("\nMacro Definition Table (MDT):")
        for name, definition in self.macro_definition_table.items():
            print(f"Macro Name: {name}")
            for line in definition:
                print(f"\t{line}")

source_code = [
    "MACRO",
    "INCR &ARG1",
    "ADD &ARG1, ONE",
    "MEND",
    "MACRO",
    "DECR &ARG1",
    "SUB &ARG1, ONE",
    "MEND",
    "START",
    "INCR A",
    "DECR B",
    "END",
]

macro_processor = MacroProcessor()

macro_processor.process_pass1(source_code)
macro_processor.display_tables()

expanded_code = macro_processor.process_pass2(source_code)

print("\nExpanded Source Code (Pass 2):")
for line in expanded_code:
    print(line)
```

Output



```
$ python3 Code-A2.py
Macro Name Table (MNT):
Macro Name: INCR, MDT Index: 0
Macro Name: DECR, MDT Index: 2

Macro Definition Table (MDT):
Macro Name: INCR
        INCR &ARG1
        ADD &ARG1, ONE
Macro Name: DECR
        DECR &ARG1
        SUB &ARG1, ONE

Expanded Source Code (Pass 2):
START
INCR A
ADD A, ONE
DECR B
SUB B, ONE
END
```