|  |  |
| --- | --- |
| **Ex No: 3** | **Simulation of Macroprocessors** |
| **Date of Exercise** | **20/01/2022** |

|  |
| --- |
| **AIM** |
| To write a program to simulate the functions of a macroprocessor. |
| **DESCRIPTION** |
| A macroinstruction is a notational convenience for the programmer. It allows the programmer to write shorthand version of a program (module programming). The macro processor replaces each macro invocation with the corresponding sequence of statements (expanding)  **Macro Processor**   * Recognize macro definitions * Save the macro definition * Recognize macro calls * Expand macro calls |
| **ALGORITHM** |
| 1. Get the C program, which include the macro definition.  2. Expand the macro by replacing the macro call with macro body and while expanding replace the formal parameter with actual parameter.  3. Print the expanded output. |
| **SAMPLE INPUT & OUTPUT** |
| Enter the program  #define ADD(X) (X=X/X)  #include<stdio.h>  void main()  {  int a = 10;  ADD(a);  }  The Expanded program is    #include<stdio.h>  void main()  {  int a = 10;  a=a/a; // Macro Expansion  } |
| **QUESTION SET** |
| 1. Write a program to simulate the functions of a Macroprocessor. |

**PROGRAM:**

with open('/content/input\_ex3.txt') as f:

source = f.read()

lines=source.splitlines()

fun\_names=[]

fun\_bodies=[]

for line in lines:

if'#define' in line:

line=line.split(' ')

fun\_name=line[1][:line[1].find('(')]

fun\_names.append(fun\_name)

fun\_bodies.append(line[2])

d=dict(zip(fun\_names,fun\_bodies))

for line in lines:

if'#define' not in line:

for name in fun\_names:

if name in line:

line=line.replace(name,'').replace('(','').replace(')','')

line=line.replace(';','')

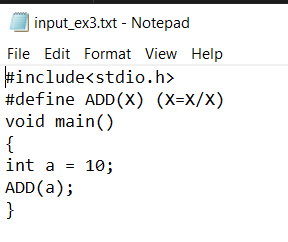
line=d[name].replace('X',line[0])+';'

line=line.replace('(','').replace(')','')

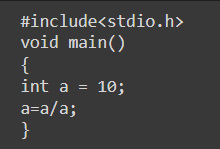
print(line)

**Output Screenshot**

Original File:



File after code execution:



**Result**

Thus the program ran successfully and the output was verified.