**COA LAB**

**Experiment – 6**

**Write an assembly language program for the following problem statements:**

1. A routine that pushes numbers 6 and 4 on top of the stack, then pops the two numbers one by one from stack, add them and pushes the result back to the top of stack.
2. First loop places 15 numbers from 1-15 on top of stack using push instruction. In the second loop, use the pop instruction to pop two numbers from the top of the stack, add them, and push the result back to the stack. Repeat the second loop until only one number is left on top of the stack.

**Program no. 1**

**Algorithm:**

1. Mark the start of the frame
2. Create a label ‘pushpopadd’ to represent the routine name.
3. Call routine ‘pushpopadd’.
4. Place the halt instruction below the call instruction and move the label ‘pushpopadd’ below the halt instruction.
5. Inside the label/routine ‘pushpopadd’, push numbers 6 and 4 on top of stack, then pop the two numbers one by one from stack, add them and push the result back to the top of stack.
6. Pop to any general-purpose register to get the return address on top of the stack
7. Returns to Halt instruction and exit.

**Assembly Language code:**

**MSF** *;Main start function*

**CAL $pushpopadd** *;Call the function pushpopadd*

**HLT** *;Halt the program execution*

**pushpopadd:** *;Define the function pushpopadd*

**PSH #6** *;Push the value 6 onto the stack*

**PSH #4** *;Push the value 4 onto the stack*

**POP R01** *;Pop the top value from the stack into register R01*

**POP R02** *;Pop the second value from the stack into register R02*

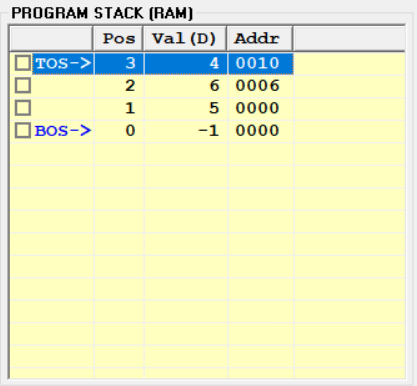
**ADD R01, R02** *;Add the values in R01 and R02 and store the result in R02*

**PSH R02** *;Push the result onto the stack*

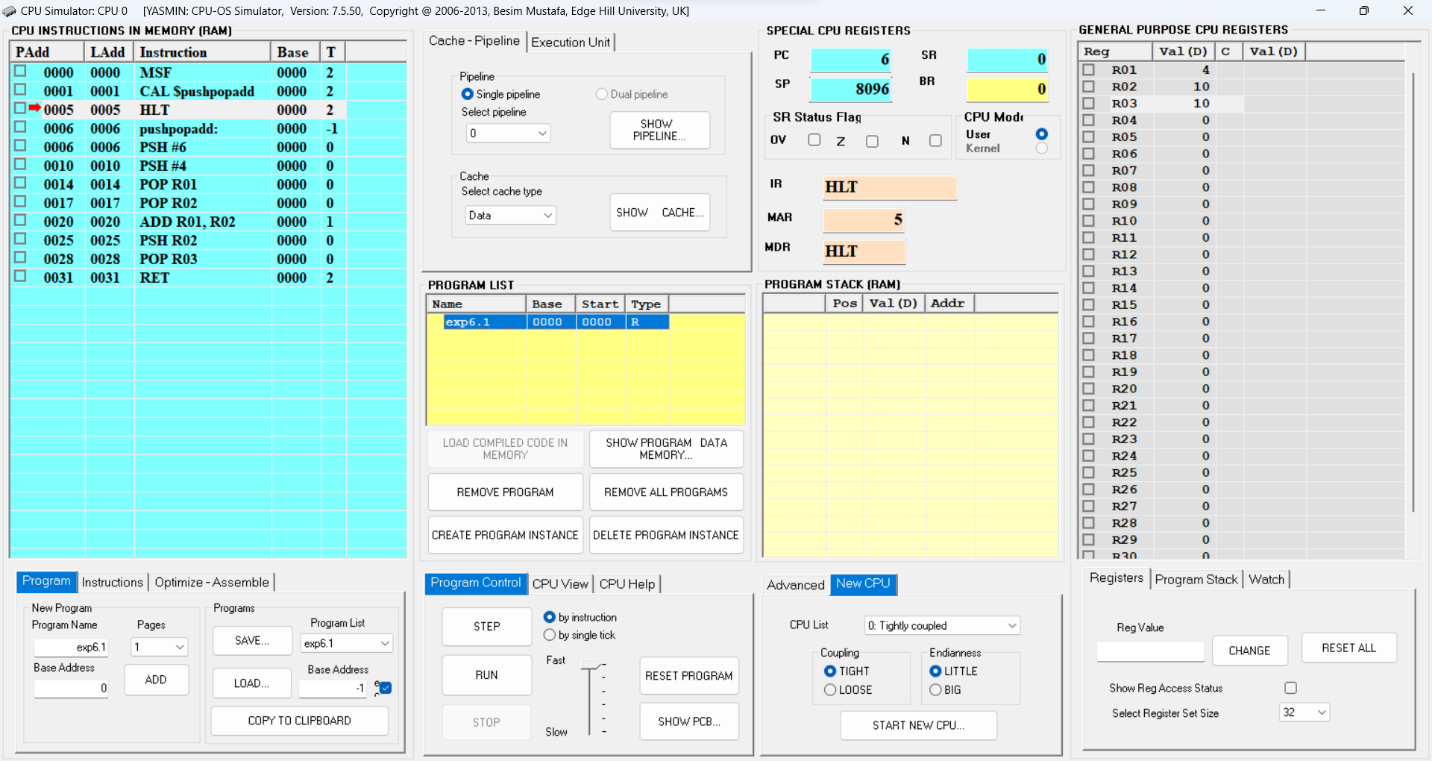
**POP R03** *;Pop the result from the stack into register R03*

**RET** *;Return from the function*

**Result:**

****

**Fig-6.1.a:** Program Stack window after pushing all the values

****

**Fig-6.1.b:** CPU Simulator Window

**Program no. 2**

**Algorithm:**

1. Initialize register R01 to 1 to hold the value of the first number to be pushed onto the stack.
2. Enter a loop to push 15 numbers onto the stack using the PUSH instruction, incrementing R01 each time to hold the value of the next number.
3. After the first loop completes, enter a second loop to pop two numbers from the top of the stack, add them, and push the result back to the stack.
4. Repeat the second loop until only one number is left on top of the stack.
5. The final result will be the value left on top of the stack.
6. Halt the program after the second loop completes.

**Assembly Language code:**

**MOV #1, R01** *;Initialize register R01 to 1*

**MSF** *;Main start function*

**CAL $pushpoploop** *;Call the function pushpoploop*

**HLT** *;Halt the program execution*

**pushpoploop:** *;Define the function pushpoploop*

**PSH R01** *;Push the value of register R01 onto the stack*

**INC R01** *;Incrementing value of R01 by 1*

**CMP #15, R01** *;Compare value of R01 with 15*

**JLE $pushpoploop** *;Jump back to pushpoploop if R01 <= 15*

**addloop** *;Define the function* addloop

**POP R01** *;Pop the top value from the stack into register R01*

**POP R02** *;Pop the second value from the stack into register R02*

**ADD R01, R02** *;Add the values in R01 and R02 and store the result in R02*

**PSH R02** *;Push the value of register R02 onto the stack*

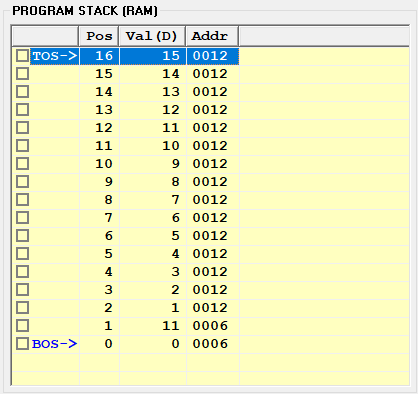
**CMP #120, R02** *;Compare value of R02 with 120*

**JNE $addloop** *;Jump back to addloop if the result is not equal to 120*

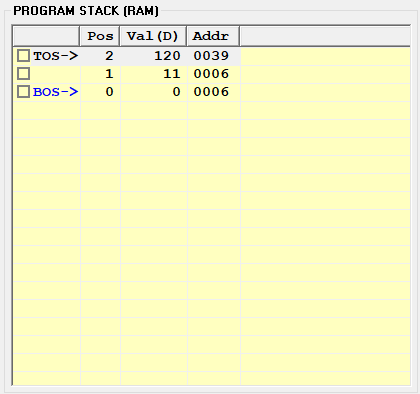
**POP R03** *;Pop the final result from the top of the stack*

**RET** *;* *Return from the subroutine*

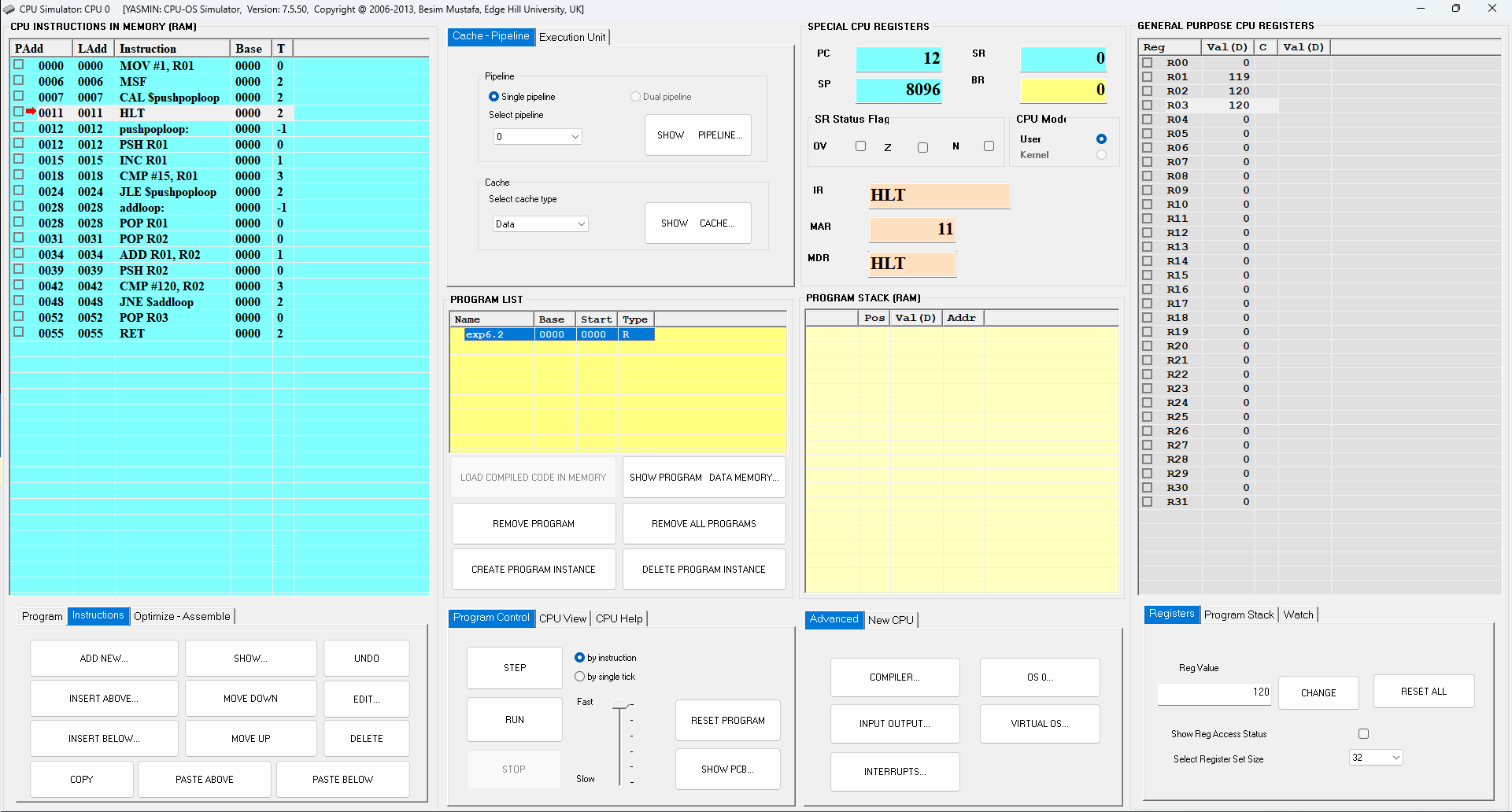
**Result:**



**Fig-6.2.a:** Program Stack window before exiting the first loop



**Fig-6.2.b:** Program Stack window before exiting the second loop



**Fig-6.2.c:** CPU Simulator Window