

# Bahria University,

## Karachi Campus



LAB EXPERIMENT NO.

03

### LIST OF TASKS

TASK NO	OBJECTIVE
01	Write a program that takes user input and then print counting from 0 till this number.
02	Write a program will take an input number from the user as the limit and then calculate the sum of all numbers starting from 1 up to the specified limit.
03	Write a program that print the series of odd numbers in the range of 0-10.
04	Write a program that takes any hardcoded value input from user and then print the table of it .
05	Write a program that takes two numbers as an input and perform multiplication iterative addition.

Submitted On:  
**Date: 26-Sep-2025**

**Task No. 01:**

Write a program that takes user input and then print counting from 0 till this number

**Solution:**

<i>VVM Assembly Language</i>	<i>VVM Machine Language</i>
IN	901
STO 30	330
LDA 31	531
STO 31	331
OUT	902
LDA 31	531
SUB 30	230
BRZ 12	712
LDA 31	531
ADD 32	132
STO 31	331
BR 03	603
HLT	000
*30	*30
DAT 000	000
*31	*31
Dat 000	000
*32	*32
DAT 001	001

**Output:**

**I/O Log**

```

[00] Begin Run
[00] Input: 4
[04] Output: 0
[04] Output: 1
[04] Output: 2
[04] Output: 3
[04] Output: 4
[12] End - Normal

```

**Hardware View**

**PROCESSOR**

Accum. 000

Prog. Ctr. 12

Instr. Reg. HLT

Instruction Cycle: 45

**RAM**

	_0_	_1_	_2_	_3_	_4_	_5_	_6_	_7_	_8_	_9_
0_	901	330	531	331	902	531	230	712	531	132
1_	331	603	000	000	000	000	000	000	000	000
2_	000	000	000	000	000	000	000	000	000	000
3_	004	004	001	000	000	000	000	000	000	000
4_	000	000	000	000	000	000	000	000	000	000
5_	000	000	000	000	000	000	000	000	000	000
6_	000	000	000	000	000	000	000	000	000	000
7_	000	000	000	000	000	000	000	000	000	000
8_	000	000	000	000	000	000	000	000	000	000
9_	000	000	000	000	000	000	000	000	000	000

**Execute**

Resume Step Pause

Restart S speed F

☐ Show Source Window ☒ Tick

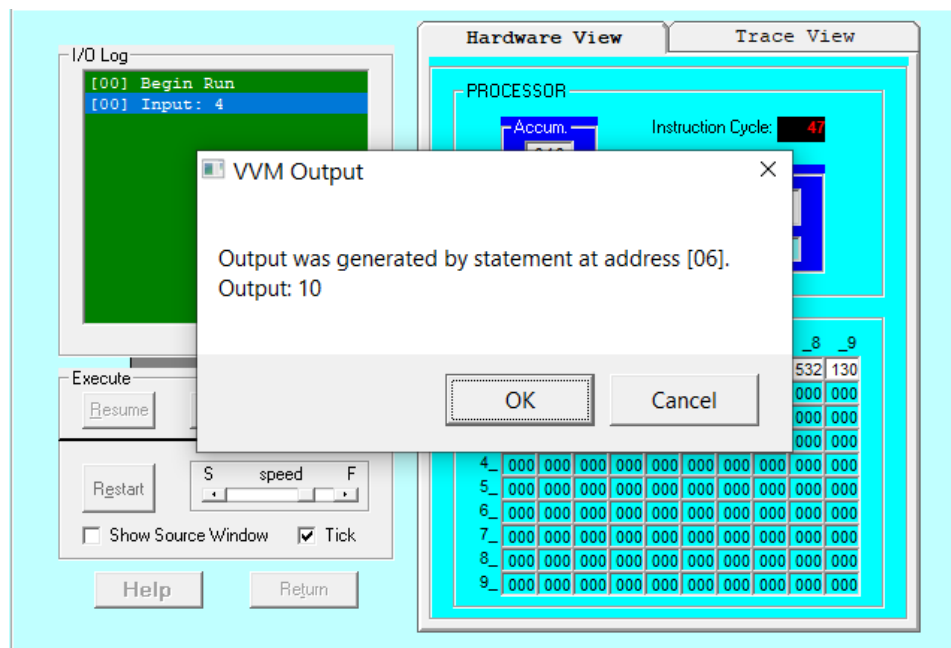
Help Return

**Task No. 02:**

Write a program will take an input number from the user as the limit and then calculate the sum of all numbers starting from 1 up to the specified limit.

**Solution:**

<i>VVM Assembly Language</i>	<i>VVM Machine Language</i>
IN	901
STO 31	331
LDA 31	531
SUB 30	230
BRP 08	808
LDA 32	532
OUT	902
HLT	000
LDA 32	532
ADD 30	130
STO 32	332
LDA 30	530
ADD 33	133
STO 30	330
BR 02	602
*30	*30
DAT 001	001
*32	*32
DAT 000	000
*33	*33
DAT 001	001

**Output:**

**Task No. 03:**

Write a program that print the series of odd numbers in the range of 0-10.

**Solution:**

<i>VVM Assembly Language</i>	<i>VVM Machine Language</i>
LDA 31	231
SUB 30	130
BRP 04	704
HLT	000
LDA 30	130
OUT	902
ADD 32	332
STO 30	330
BR 00	700
*30	*30
DAT 001	001
*31	*31
DAT 010	010
*32	*32
DAT 002	002

**Output:**

The screenshot displays the VVM (Visual Basic Virtual Machine) interface during the execution of a program. The interface is divided into several windows:

- I/O Log:** Shows the execution log with the following entries:
  - [00] Begin Run
  - [05] Output: 1
  - [05] Output: 3
  - [05] Output: 5
  - [05] Output: 7
  - [05] Output: 9
  - [03] End - Normal
- Hardware View:** Displays the internal state of the processor:
  - Accum. (Accumulator):** -001
  - Instr. Reg. (Instruction Register):** 000
  - Prog. Ctr. (Program Counter):** 03
  - Instruction:** HLT
  - Instruction Cycle:** 44
- RAM:** Shows a memory dump with 10 rows (0-9) and 10 columns (\_0-\_9). The values are:
 

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9
0	531	230	804	000	530	902	132	330	600	000
1	000	000	000	000	000	000	000	000	000	000
2	000	000	000	000	000	000	000	000	000	000
3	011	010	002	000	000	000	000	000	000	000
4	000	000	000	000	000	000	000	000	000	000
5	000	000	000	000	000	000	000	000	000	000
6	000	000	000	000	000	000	000	000	000	000
7	000	000	000	000	000	000	000	000	000	000
8	000	000	000	000	000	000	000	000	000	000
9	000	000	000	000	000	000	000	000	000	000

The bottom control panel includes buttons for Run, Step, Pause, Restart, and Help, along with a speed slider and checkboxes for Show Source Window and Tick.

**Task No. 04:**

Write a program that takes any hardcoded value input from user and then print the table of it.

**Solution:**

<i>VVM Assembly Language</i>	<i>VVM Machine Language</i>
IN	901
STO 33	333
LDA 34	234
STO 30	330
LDA 35	235
STO 31	331
LDA 33	233
STO 36	336
LDA 36	236
OUT	133
LDA 36	336
ADD 33	230
STO 36	234
LDA 30	330
ADD 34	230
STO 30	131
LDA 30	722
SUB 31	708
BRP 22	000
BR 08	*30
HLT	000
*30	*31
DAT 000	000
*31	*32
DAT 000	000
*32	*33
DAT 000	000
*33	*34
DAT 000	001
*34	*35
DAT 000	011
*35	*36
DAT 011	000
*36	
DAT 000	

**Output:**

The screenshot displays a VVM (Virtual Machine) interface with three main panels:

- I/O Log:** A list of input and output operations. The last entry is "[22] End - Normal".
- Execute:** A section with buttons for "Run", "Step", and "Pause". Below these are "Restart", a "speed" slider between "S" and "F", and checkboxes for "Show Source Window" (unchecked) and "Tick" (checked). At the bottom are "Help" and "Return" buttons.
- Hardware View:** A section showing the internal state of the processor and RAM.
  - PROCESSOR:**
    - Accum. (Accumulator): 000
    - Prog. Ctr. (Program Counter): 22
    - Instr. Reg. (Instruction Register): 00, HLT
    - Instruction Cycle: 128
  - RAM:** A table showing memory contents for addresses 0 to 9. Each address has 10 data bytes.

**RAM Data Table:**

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9
0_	901	333	534	330	535	331	533	336	536	902
1_	536	133	336	530	134	330	530	231	822	608
2_	000	000	000	000	000	000	000	000	000	000
3_	011	011	000	008	001	011	088	000	000	000
4_	000	000	000	000	000	000	000	000	000	000
5_	000	000	000	000	000	000	000	000	000	000
6_	000	000	000	000	000	000	000	000	000	000
7_	000	000	000	000	000	000	000	000	000	000
8_	000	000	000	000	000	000	000	000	000	000
9_	000	000	000	000	000	000	000	000	000	000

**Task No. 05:**

Write a program that takes two numbers as an input and perform multiplication iterative addition.

**Solution :**

<i>VVM Assembly Language</i>	<i>VVM Machine Language</i>
IN	901
STO 30	330
IN	901
STO 31	331
LDA 30	230
SUB 32	332
BRP 10	710
LDA 34	234
OUT	902
HLT	234
LDA 34	131
ADD 31	334
STO 34	232
LDA 32	333
ADD 33	332
STO 32	704
BR 04	000
*30	*30
DAT 000	000
*31	*31
DAT 000	000
*32	*32
DAT 001	001
*33	*33
DAT 001	001
*34	*34
DAT 000	000

**Output:**