National University of Computer and Emerging Sciences

Lab Manual

Computer Organization and Assembly Language

Lab 05

Instructor	Sarosh Humayun, Haiqa Saman
Class	CS3
Sections	D
Semester	Fall 2023

Fast School of Computing

FAST-NU, Lahore, Pakistan

ACTIVITY 1:

Initialize AX with last 4 digits of your roll number as **Hexadecimal number** (for example, if your roll number is 16L-4195 then AX should be initialized with 0x4195). Write a subroutine which receives AX as input and returns a number of 1s in AX.

$$n = binary\_ones(Roll\ \#)$$

For example, # of 1s in 0x 4195 is

$$n = ones(0x4195) = ones(0100\_0001\_1001\_0101) = 6$$

ACTIVITY 2:

Following table shows a number pyramid (we call it Al-Khwarizmi Pyramid). This pyramid is expanding based on the value of s, its size.

Write a program which uses $s = n + 5$ (n from **Activity 2**) as the size of Al-Khwarizmi Pyramid and returns the cumulative sum. For example, if $n = 6$ then $s = 11$, and the program should return **506**.

Size (s)	Al-Khwarizmi Pyramid																				Cumulative Sum
1										1											**1**
2										1	2	1									**5**
3									1	2	3	2	1								**14**
4								1	2	3	4	3	2	1							**30**
5							1	2	3	4	5	4	3	2	1						**55**
6						1	2	3	4	5	6	5	4	3	2	1					**91**
7					1	2	3	4	5	6	7	6	5	4	3	2	1				**140**
8				1	2	3	4	5	6	7	8	7	6	5	4	3	2	1			**204**
9			1	2	3	4	5	6	7	8	9	8	7	6	5	4	3	2	1		**285**
10		1	2	3	4	5	6	7	8	9	10	9	8	7	6	5	4	3	2	1	**385**
11	1	2	3	4	5	6	7	8	9	10	11	10	9	8	7	6	5	4	3	2	1	**506**

Practice (Unmarked):

Question 1:

Difference of two sets ($S_1 - S_2$) is a set having elements of S_1 which are NOT Present in S_2, see following examples for detail. Your task is to write a subroutine in Assembly Language that finds Difference of two sets (S_1-S_2). Note that both the sets are sorted and have distinct elements only.

Example 1	Example 2
S_1: -3, -1, 2, 5, 6, 8, 9 S_2: -2, 2, 6, 7, 9 **Difference**: -3, -1, 5, 8	S_1: -3, -1, 2, 5, 6, 8, 9 S_2: 1, 3, 7 **Difference**: -3, -1, 2, 5, 6, 8, 9

Question 2:

Write two subroutines for 16-bit multiplication and 32-bit addition to solve the following problem from Lab4:

Initialize AX with last 4 digits of your roll number (for example, if your roll number is 16L-1105 then AX should be initialized with 1105). Store $\overline{AX}$ in BX. Make a 32-bit memory variable f, initialize it with 0 and compute

$$f = (A \times B) + \{A, B\}$$

$\times$ is **Multiplication** operation, $+$ is **Addition** operation whereas $\{A, B\}$ **concatenates** 16-bit **A** and **B** to form **32-bit** number.