COL 216 : Assignment 1: Area Under a Curve

Ramneet Singh | Mustafa Chasmai
2019CS50445 | 2019CS10341

July 4, 2021

Design Characteristics:

1. Error handling in Input
   - MIPS does not have a built-in error handling mechanism.
   - Normal integer input allows special characters, string and also no input.
   - To restrict coordinates to integers, input taken as string and parsed to get integers.
2. Avoiding Overflow as much as possible
   - Integer arithmetic leads to overflows when the result is larger than 32 bits.
   - To solve this issue, integer coordinates were converted to doubles and further operations were performed as double operations.

Design Choices:

1. Assumed number of points to be ≥ 1 as without that, the problem does not make mathematical sense. (There would be no curve if there are ≤ 0 points). For such an input, relevant error message is displayed and execution stopped.
2. Sequential input taken for the points, with prompts for the x-coordinate first, followed by that for the y-coordinate, for each input points.
3. Coordinates not stored in memory, area contributions from each pair computed dynamically, thereby saving memory space.
4. Relevant messages displayed for every input, error, output area and end of execution.
5. Assumed each coordinate to be a 32-bit integer.
6. Assumed area under the x-axis as positive.
7. If x-coordinates of two consecutive points are equal, it is assumed that the curve considered has them in the same order as that given by the user. (a different order does not make sense)
8. For the computation of area, the expressions were manipulated to keep as few cases as possible.

Approach of Solution:

First, the number of points to be given is taken as input. Then for each input point, first the x-coordinate is taken and then the y-coordinate. The coordinates are not stored in memory (to conserve space), and the area under them is computed on the fly.

As each point is input, the area under this point and the last point is calculated. Depending on the location of the two points, the 4 major cases identified are reduced to only 2 cases and corresponding areas are computed: (the points are (x_1, y_1) and (x_2, y_2))

1. $y_1 * y_2 \geq 0$: Area $= 1/2 * (x_2 - x_1) * (|y_1| + |y_2|)$
2. $y_1 * y_2 < 0$: Area $= 1/2 * (x_2 - x_1) * (y_1^2 + y_2^2)/(|y_1| + |y_2|)$

The points are initially taken as integers and then converted to doubles using floating point registers. Subsequent calculations are performed using double arithmetic as multiplication of 32 bit coordinates can be 64 bit. A register stores the total area for multiple points, and after input of each point other than the first, the calculated new area is added to this.

As soon as all n points have been taken, the computed area is outputted and execution is stopped. Relevant messages for the same are displayed on the console.

Testing strategy:

1. **Random Testing:**
   Some random inputs are given to program and its output is checked with manual calculations.

Type	No. of Points	Points	Area
Random	3	(2, 2), (3, 3), (4, 4)	6
	4	(1, 2), (3, 7), (4, 90), (10, 17)	378.5
	2	(0, 2), (15, 0)	15
	1	(1, 1)	0
	4	(12, 2), (23, -7), (14, 90), (110, 17)	4790.342
	2	(-150, 0), (0, 200)	15000
	3	(1, 1234), (1234, 100), (2468, 72)	928535

2. **Boundary Cases:**
   The different boundary cases are identified and one test input from each such case is checked with manual calculations.

Type	Sub-Type(s)	No. of Points	Points	Area
$x_1 \geq 0$, $x_2 \geq 0$	$y_1 \geq 0$, $y_2 < 0$	2	(78, 197), (200, -5)	11727.099
	$y_1 < 0$, $y_2 \geq 0$	2	(200, -5), (379, 80)	6765.147
	$y_1 \geq 0$, $y_2 \geq 0$	2	(19, 60), (131, 23)	4648
	$y_1 < 0$, $y_2 < 0$	2	(38, -18), (40,-97)	115
$x_1 < 0$, $x_2 < 0$	$y_1 \geq 0$, $y_2 \geq 0$	2	(-113, 82), (-90, 73)	1782.5
	$y_1 < 0$, $y_2 < 0$	2	(-87, -49), (-2, -91)	5950
	$y_1 \geq 0$, $y_2 < 0$	2	(-200, 115), (-133, -18)	3412.718
	$y_1 < 0$, $y_2 \geq 0$	2	(-133, -18), (-63, 44)	1275.806
$x_1 < 0$, $x_2 \geq 0$	$y_1 \geq 0$, $y_2 \geq 0$	2	(-311, 12), (17, 290)	49528
	$y_1 < 0$, $y_2 \geq 0$	2	(-291, -171), (90, 71)	26986.4504
	$y_1 \geq 0$, $y_2 < 0$	2	(-213, 81), (2, -87)	9041.5178
	$y_1 < 0$, $y_2 < 0$	2	(-191, -78), (111, -3)	12231
$x_1 = x_2$	$y_1 \neq y_2$	2	(97, 289), (97, -1)	0
	$y_1 = y_2$	2	(53, -17), (53, -17)	0
$y_1 = y_2$		2	(-271, 13), (68, 13)	4407

3. **Ultimate Test Cases:**
   Test cases that include all the major boundary cases and hard computations, along with a large number of inputs.

No. of Points	Points	Area
10	(-5, -7), (-4, 11), (-3, 2), (-2, -10), (-1, -5), (1, -10), (2, 7), (3, 12), (4, -1), (5, -6)	61.015
10	(-10, -7), (-8, 11), (-6, 2), (-4, -10), (-2, -5), (2, 12), (4, 7), (6, -10), (8, -1), (10, -6)	111.758
10	(-12, -17), (-11, 21), (-7, 12), (-3, -1), (-1, 15), (1, -9), (3, 7), (12, 12), (24, -1), (25, -6)	288.836
10	(-25927, -79023), (-14237, 11000), (-52341, 239233), (-20, -1), (-19, 5000), (12012, 123456), (25349, 79034), (32127, 129834), (48155, -10098), (50998, -65903)	5814463841.312

4. **Invalid inputs:**
   Some cases where inputs may not be as per specifications are identified and checked for errors or wrong outputs.

Type	No. of Points	Points	Area
Inputs are not Integers	- (no input)	-	Error: Please input an integer
	2	(2, 3), (4, abc)	Error: Please input an integer
	2	(2, 3), (;', 3)	Error: Please input an integer
	2.3	-	Error: Please input an integer
	2	(2, 3), (4.12, 2)	Error: Please input an integer
	2	(2, 3), (3, two)	Error: Please input an integer
Number of points ≤ 0	0	-	Error: no of points must be > 0
	-2	-	Error: no of points must be > 0