ASSIGNMENT-1

COL 216, Spring 2021

Area Under Curve using MIPS

Jitender Kumar Yadav, 2019CS10361
Asha Ram Meena, 2019CS10337

February 23, 2021

1 Problem Statement

Write a MIPS Assembly Program for obtaining the area under a curve formed by joining successive points by a straight line.

2 Algorithm and Approach

If we join two points by a straight line, the area between the points and the x-axis is essentially a trapezium if the points lie on the same side of the x-axis, with area given by:

$$\Delta = \left| \frac{1}{2} \times (x_2 - x_1) \times (y_2 + y_1) \right|$$

If the points lie on the opposite sides of the axis, the area has to be calculated by considering the triangles which can be found using the following formula:

$$\Delta = \frac{1}{2} \times \left| \frac{x_2 - x_1}{y_2 - y_1} \right| \times (y_1 \times y_1 + y_2 \times y_2)$$

The following steps are used to calculate the area:

- The integer n is read.
- The pair (x,y) are read one by one starting from first point till the last.
- When the k$^{\text{th}}$ pair is read for $k > 1$, two registers store the current and the previous coordinates.
- The calculation of area between the current and previous points is done by using the above formula, by use of conditional statements. This area is added to the value of variable representing current area.
- The value of area at the end of all iterations in returned and printed.

3 Input and Output

The input is to be taken from the user using console and the output shall be printed on the console itself.

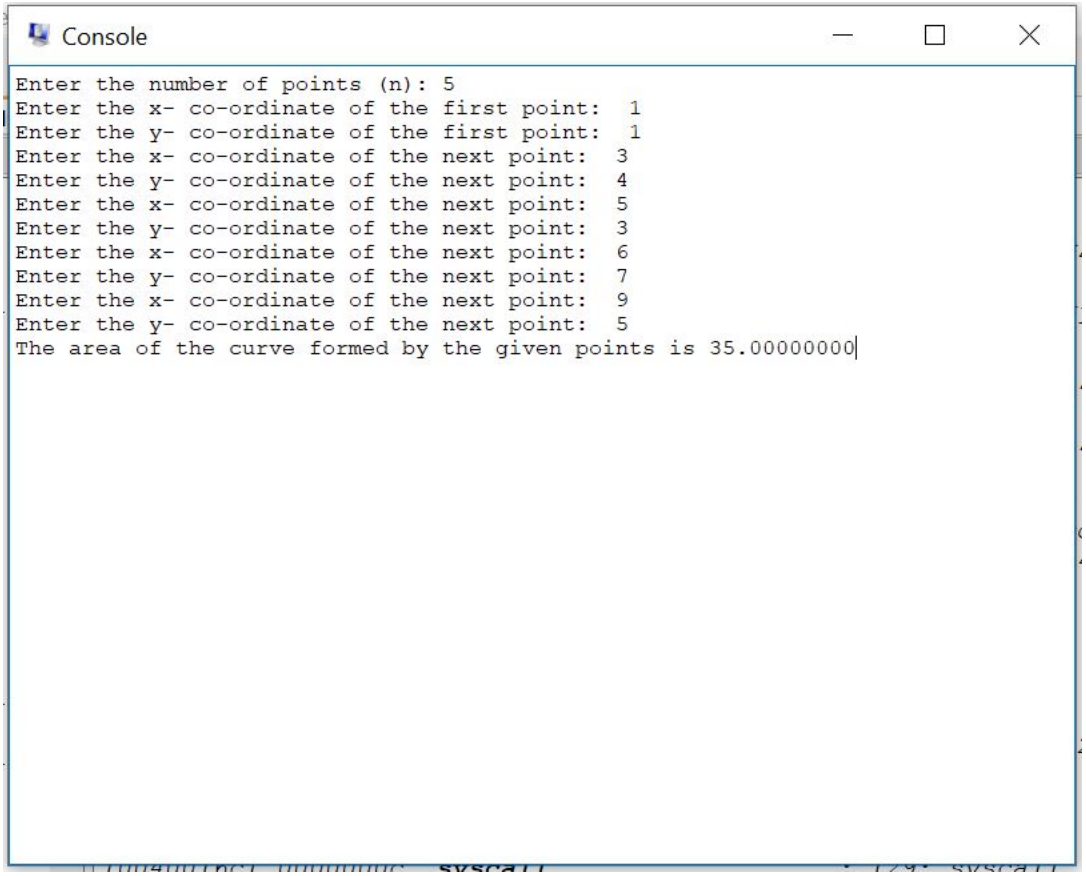
3.1 Input Specifications:

- The first line inputs an integer n, the number of points.
- The next 2n lines input the x- and y- co-ordinates of the n points one after the other for each point.
- It is assumed that $|x_i| \leq 2^{15}$ and $|y_i| \leq 2^{15}$, $1 \leq i \leq n$ where x_i and y_i represent the x- and y-coordinates of i^{th} point.

3.2 Output

- The output is calculated and shown at the end after the input has been read and processed.

3.3 Sample I/O



```
Enter the number of points (n): 5
Enter the x- co-ordinate of the first point:  1
Enter the y- co-ordinate of the first point:  1
Enter the x- co-ordinate of the next point:  3
Enter the y- co-ordinate of the next point:  4
Enter the x- co-ordinate of the next point:  5
Enter the y- co-ordinate of the next point:  3
Enter the x- co-ordinate of the next point:  6
Enter the y- co-ordinate of the next point:  7
Enter the x- co-ordinate of the next point:  9
Enter the y- co-ordinate of the next point:  5
The area of the curve formed by the given points is 35.00000000
```

4 The Code

4.1 Registers

- **$s0** : stores the number of points, n
- **$t1** : stores the x-coordinate of previous point
- **$t2** : stores the y-coordinate of previous point or its absolute value
- **$t3** : stores the iterator i to loop over n points
- **$t6** : stores the x-coordinate of current point
- **$t7** : stores the y-coordinate of current point or its absolute value
- **$t8** : stores the value of width of the trapezium ($x_2 - x_1$)
- **$t9** : stores the signed y-coordinate of the current point to be stored later in $t2
- **$t4** : stores the numerator of the area under the current line segment
- **$t5** : stores the denominator of the area under the current line segment
- **$f1** : stores the total area till the last point
- **$f2** : stores the float value of the numerator in current expression of area
- **$f3** : stores the float value of the denominator in current expression of area
- **$f4** : stores the area under the line segment joining the current and previous points

4.2 Location Labels

- **loop** : compares the value of i against n and thus inputs current point, transferring it to the correct area calculation function
- **findArea** : calculates and adds the current area to the total area register
- **pos_neg** : calculates the numerator and denominators for the current area when points are on the opposite sides
- **printArea** : prints the area to the console

5 Testing

Some test-cases were generated by a random number generator and their output was evaluated by using a python 3.0 program. While the test-cases to consider the corner cases were hand-made and evaluated.
The test cases are stored in files A1_Test1.in to A1_Test8.in and the outputs are auto-generated by a python file checker.py and stored in A1_Test_out.out in the same folder.
The code was tested with all 8 sample testcases as well as new random test cases and the final area was manually calculated. The final area came out to be the same as expected, thus validating the code stored in the file ”A1_area.asm”.