Final Project

Massih Majidi

Your task is to compute the area of a polygon given the coordinates of its vertices. You must write a function named **AreaA** in a file named "**area.asm**" that receives the number of vertices and an array of the vertex coordinates and returns the final result and a "Makefile" to build the project. Use the file "**main.c**" below. **DO NOT CHANGE IT**.
The file "main.c" uses a function called **AreaC** to calculate the area on its own. Compare the running time of your assembly code against it.

You have to submit **2 versions** of your code:

1. A **32-bit** version using the **floating-point coprocessor (x87)** instructions. (20 points)
2. A **64-bit** version using the SSE or AVX **SIMD** instructions.(80 points)

Compare the elapsed time of the C function (once normal and once compiled with "-O3") with the SIMD version.

Useful links:

SIMD:

- https://en.wikipedia.org/wiki/X86_instruction_listings#SSE2_instructions
- https://docs.oracle.com/cd/E18752_01/html/817-5477/epmpv.html
- https://sci.tuomastonteri.fi/programming/sse
- https://youtube.com/playlist?list=PL0C5C980A28FEE68D

Area Algorithem:
https://web.archive.org/web/20100405070507/http://valis.cs.uiuc.edu/~sariel/research/CG/compgeom/msg00831.html

Your code **must** comply with the following rules:

- The SIMD version must use the vectorization capability of the SIMD instructions (doing multiple operations with a single instruction).
- You must observe all C Calling Conventions across both versions.
- You must not use any global labels for communication. Every value or reference must be passed as an argument.
- The **Difference** between Your Result and sample Result must not be over 100.
- You MUST NOT PRINT ANYTHING. PRINTS ARE HANDLED BY THE GIVEN CODE. Results are checked by Script.

Remember that your code will be checked for similarity. In the case of cheating the students will receive a **negative** point. It is your responsibility to protect your code.
Please upload only the explained **".zip"** file on vc.kntu.ac.ir .

```c
#include <stdio.h>
#include <time.h>
double AreaC(int n, double list[n][2]);
double AreaA(int n, double list[n][2]);

int main(){
    int n;
    scanf("%d", &n);
    double a[n][2];
    for(int i = 0; i < n; i++) scanf("%lf%lf", &a[i][0], &a[i][1]);
    //TIME Vars
    struct timespec start, stop;
    //Get Start TIME
    clock_gettime(CLOCK_REALTIME, &start);
    double res = AreaC(n, a);
    //Get Stop TIME
    clock_gettime(CLOCK_REALTIME, &stop);
    printf("%lf\n", res);
    //Print TIME Elapsed in nano seconds
    printf("%ld\n", stop.tv_nsec  - start.tv_nsec );
    return 0;
}
double AreaC(int n,double list[n][2]){
    double area = 0;
    for(int i = 0; i < n - 1; i++)
        area += list[i][0] * list[(i+1)%n][1] - list[i][1] * list[(i+1)%n][0];
    return area > 0 ? area/2 : -area/2;
}
```

Inputs:
The first input is the number of vertices **n**, followed by **n** lines in the form of
x y
giving the coordinates of each vertex. The output is the area of the polygon.

Constraints:
x0 = 0, **y0** = 0
2 < **n** < 100
-100.0 < **x, y** < 100

Example:
Input 1:
4
0 0
0 2
3 3
3 0

Output 1:
7.500000

Input 2:
7
0 0
0 5.7
3 6.6
4 9
6 9
7 2
4 1

Output 2:
43.250000