**ENGR 313**

**Dr. Sherine Elbaradei**

**Group Project - Part 2 (Report)**

**By:**

**Ahmed A. Agiza (900121143)**

**Mohammed R. Anany (900120267)**

**Ola Amr Hussein (900120585)**

**Rawan Abbas (900120743)**

**Pseudo-code:**

*Trapezoidal rule:*

*function trapezoidal(SamplePoints points, Number n):*

*Number result = 0*

*check if the data are equally spaced*

*result = points[0].x + points[n - 1].x*

*for each point from 1 to n-2*

*result = result + 2 \* point.y*

*result = result \* (points[n – 1].x – points[0].x(*

*result = result / (2 \* n)*

*return result*

*endfunction*

*Simpson 1/3 rule:*

*function simpson1\_3(SamplePoints points, Number n):*

*Number result = 0*

*check if the data are equally spaced*

*check if the number of segements is even*

*result = points[0].x + points[n - 1].x*

*for each odd point from 1 to n-2*

*result = result + 4 \* point.y*

*for each even point from 1 to n-2*

*result = result + 2 \* point.y*

*result = result \* (points[n – 1].x – points[0].x)*

*result = result / (3 \* n)*

*return result*

*endfunction*

*Simpson 3/8 rule:*

*function simpson3\_8 (SamplePoints point, Number n):*

*Number result = 0*

*check if the data are equally spaced*

*check if the number of segements is a multiple of 3*

*result = points[0].x + points[n - 1].x*

*for each point belonging to 1, 4, 7..*

*result = result + 3 \* point.y*

*for each point belonging to 2, 5, 8..*

*result = result + 3 \* point.y*

*for each point belonging to 3, 6, 9..*

*result = result + 2 \* point.y*

*result = result \* (points[n – 1].x – points[0].x)*

*result = result / 8*

*return result*

*endfunction*

**Flowcharts:**

|  |  |
| --- | --- |
| Trapezoidal |  |
| Simpson’s 1/3 |  |
| Simpson’s 3/8 |  |

**Computer Programs:**

C++ files are attached, definitions of variables, functions and procedures are outlined through the comments of the source code.

*Main functions:*

***trapezoidal(Point \*points, int ni, int nf)*** *: computes the integral from sample points using trapezoidal rule.*

***simpson13(Point \*points, int ni, int nf)*** *: computes the integral from sample points using Simpson’s 1/3 rule.*

***simpson38(Point \*points, int ni, int nf)*** *: computes the integral from sample points using Simpson’s 3/8 rule.*

***computeWithBestMethod(Point \*points, int ni, int nf)*** *: computes the integral from sample points in certain interval using the appropriate method.*

***getIntegral(Point \*points, int ni, int nf)*** *: computes the integral from sample points by dividing them to subinterval and calculating the integral using a composition of different methods.*

***getMultipleIntegral(Point3D \*points, int w, int h, int xi, int xf, int yi, int yf, int seg)*** *: computes the multiple integral from sample points of 3D function by integrating along the y-axis at different x-values using the appropriate methods then integrating the results along the x-axis.*

**Test Results:**

*#1*

*Calculated integral: 0.791282*

*Exact integral: 0.79124*

*Relative error: ~0.00520838%*

*#2*

*Calculated integral: 2816*

*Exact integral: 2816*

*Integral relative error: 0%*

*Calculated average temperature: 58.6667*

*Exact average temperature: 58.6667*

*Temperature relative error: ~5.68182e-006%*