PROGRAMMING ASSIGNMENT 3

Subject: Loops and functions
Due date: 19.11.2014

1. Mathematical Background

Integration is a very valuable part of engineering mathematics. However, not all the integrals are computable. On the other hand, integration calculation has a computation cost. In those kinds of problems, we may utilize numerical methods. According to [1] a perfect example of if this has been given below:

$$\int_0^2 e^{x^2}\,dx$$

We now need to talk a little bit about estimating values of definite integrals.

$$\int_a^b f(x)\,dx$$

by thinking of the integral as an area problem and using known shapes to estimate the area under the curve. Let's get first develop the methods and then we'll try to estimate the integral shown above.

1.1 Midpoint Rule

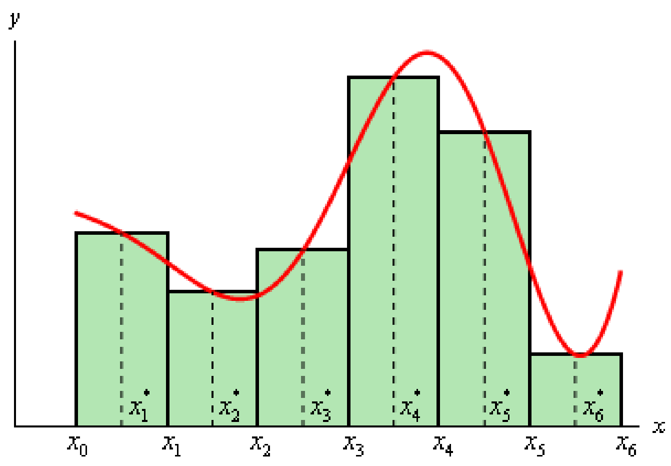
This is the rule that should be somewhat familiar to you. We will divide the interval $[a,b]$ into n subintervals of equal width,

$$\Delta x = \frac{b-a}{n}$$

We will denote each of the intervals as follows,

$$[x_0, x_1], [x_1, x_2], \ldots, [x_{n-1}, x_n] \quad \text{where } x_0 = a \text{ and } x_n = b$$

Then for each interval let x_i^* be the midpoint of the interval. We then sketch in rectangles for each subinterval with a height of $f(x_i^*)$. Here is a graph showing the set up using $n = 6$.



We can easily find the area for each of these rectangles and so for a general n we get that,

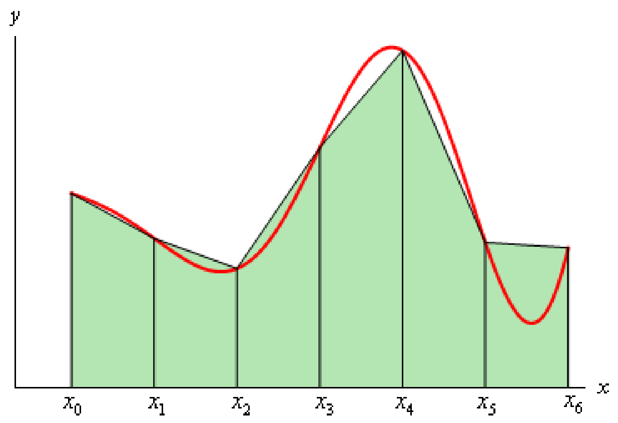
$$\int_a^b f(x)\,dx \approx \Delta x\, f\left(x_1^*\right) + \Delta x\, f\left(x_2^*\right) + \cdots + \Delta x\, f\left(x_n^*\right)$$

1.2 Trapezoid Rule

For this rule we will do the same set up as for the Midpoint Rule. We will break up the interval $[a,b]$ into n subintervals of width,

$$\Delta x = \frac{b-a}{n}$$

Then on each subinterval we will approximate the function with a straight line that is equal to the function values at either endpoint of the interval. Here is a sketch of this case for $n=6$.



Each of these objects is a trapezoid (hence the rule's name…) and as we can see some of them do a very good job of approximating the actual area under the curve and others don't do such a good job.

The area of the trapezoid in the interval $\left[x_{i-1}, x_i\right]$ is given by,

$$A_i = \frac{\Delta x}{2}\left(f\left(x_{i-1}\right) + f\left(x_i\right)\right)$$

So, if we use n subintervals the integral is approximately,

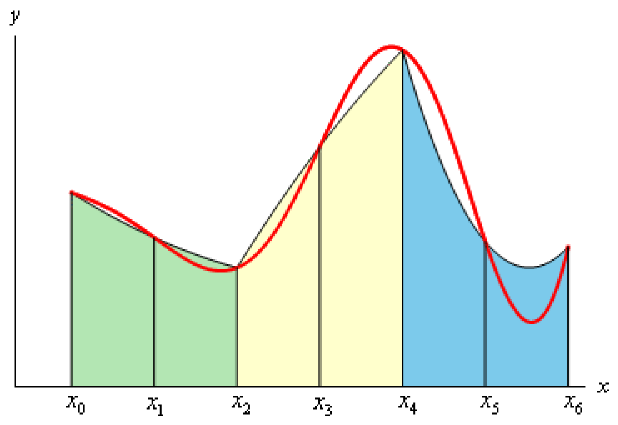
$$\int_a^b f(x)\,dx \approx \frac{\Delta x}{2}\left(f(x_0)+f(x_1)\right)+\frac{\Delta x}{2}\left(f(x_1)+f(x_2)\right)+\cdots+\frac{\Delta x}{2}\left(f(x_{n-1})+f(x_n)\right)$$

1.3 Simpson's Rule

This is the final method we're going to take a look at and in this case we will again divide up the interval [a,b] into n subintervals. However unlike the previous two methods we need to require that n be even. The reason for this will be evident in a bit. The width of each subinterval is,

$$\Delta x = \frac{b-a}{n}$$

In the Trapezoid Rule we approximated the curve with a straight line. For Simpson's Rule we are going to approximate the function with a quadratic and we're going to require that the quadratic agree with three of the points from our subintervals. Below is a sketch of this using $n=6$. Each of the approximations is colored differently so we can see how they actually work.



Notice that each approximation actually covers two of the subintervals. This is the reason for requiring n to be even. Some of the approximations look more like a line than a quadratic, but they really are quadratics. Also note that some of the approximations do a better job than others. It can be shown that the area under the approximation on the intervals $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$ is,

$$A_i = \frac{\Delta x}{3}\left(f(x_{i-1})+4f(x_i)+f(x_{i+1})\right)$$

If we use n subintervals the integral is then approximately,

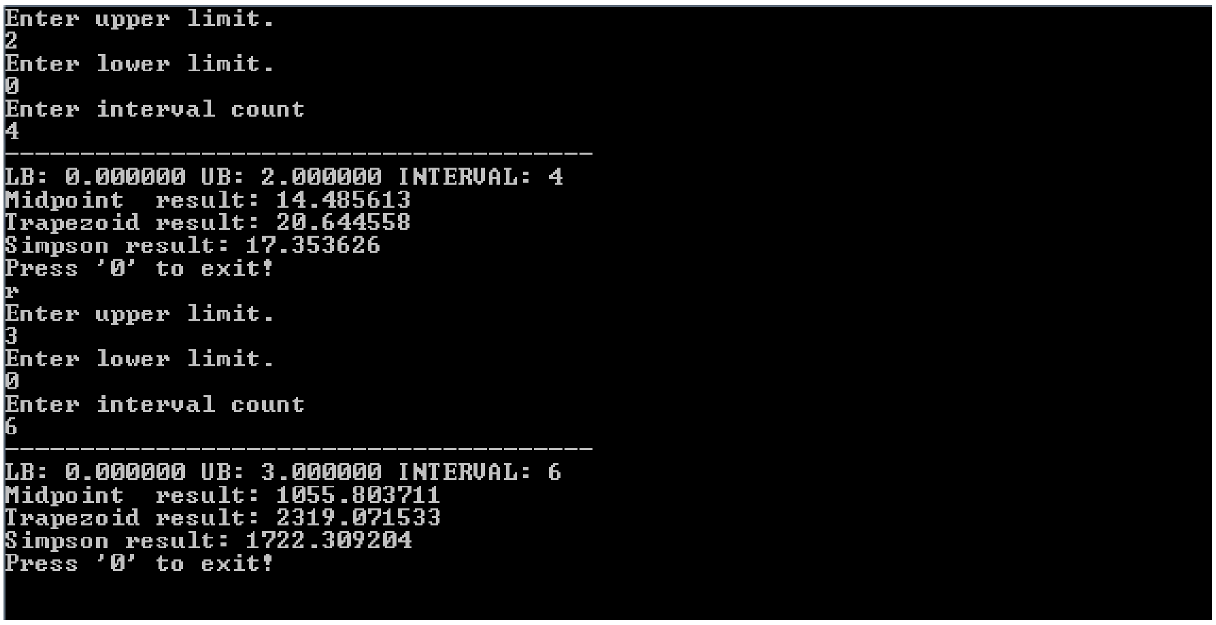
$$\int_a^b f(x)\,dx \approx \frac{\Delta x}{3}\left(f(x_0)+4f(x_1)+f(x_2)\right)+\frac{\Delta x}{3}\left(f(x_2)+4f(x_3)+f(x_4)\right)$$
$$+\cdots+\frac{\Delta x}{3}\left(f(x_{n-2})+4f(x_{n-1})+f(x_n)\right)$$

If we increment the interval number we better approximate the result.

Part I – Integration computation by numerical methods

In this part, you are assigned to prepare a console application which expects upper bound and lower bound of integral as well as interval count. Your module must calculate the integration by three methods listed above. Furthermore, your program must continue to work till the user inputs '0' character.

Your fundamental functions will be f(x)= e^{x^2} and f(x)= $2e^x - 2x$.



```
Enter upper limit.
2
Enter lower limit.
0
Enter interval count
4
----------------------------------------
LB: 0.000000 UB: 2.000000 INTERVAL: 4
Midpoint  result: 14.485613
Trapezoid result: 20.644558
Simpson result: 17.353626
Press '0' to exit!
r
Enter upper limit.
3
Enter lower limit.
0
Enter interval count
6
----------------------------------------
LB: 0.000000 UB: 3.000000 INTERVAL: 6
Midpoint  result: 1055.803711
Trapezoid result: 2319.071533
Simpson result: 1722.309204
Press '0' to exit!
```

Fig.1 A sample screenshot based on function f(x)= e^{x^2}

This screenshot has been made for only one function. Therefore your program must show two of the function results in one time.

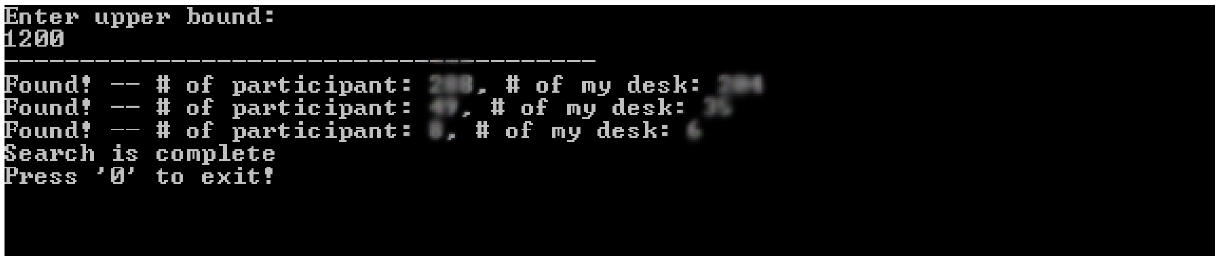
Part II – The Game "Where was I?"

Suppose that there exists a meeting and we do not know how many people attend to it. However, we see a man quitting from the meeting. If we ask the man how many people exist at the meeting he just replies "I do not know". Nonetheless he says "I was sitting on a chair numbered *x* and summation of the numbers below and above me equals. In other words, if the number of the chair is x, then 1,2,3….x-1 = x+1, x+2,…...t-1,t (total participant count). Besides, we do not know the value of *t* and the chair number *x*.

Total participant 8, and the chair number 6 is an example configuration. Because 1+2+3+4+5 = 7+8 = 15. Your task is to find similar configurations by one by one try-out.

This problem cannot be solved by typical mathematical equations. Therefore we need to write a program which uses brute-force approach.

In this part your task is to write a program which asks the upper bound of the participant number and executes your algorithm to find the total participant count and the chair number.



```
Enter upper bound:
1200
---------------------------------------
Found! -- # of participant: [redacted], # of my desk: [redacted]
Found! -- # of participant: [redacted], # of my desk: [redacted]
Found! -- # of participant: [redacted], # of my desk: [redacted]
Search is complete
Press '0' to exit!
```

Fig.2 A sample screenshot taken for upper bound limit = 1200

<u>Your program must implement a recursive function design.</u>

Notes

- Don't miss the deadline.
- Save all your work until the assignment is graded.
- The assignment must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.
- You can ask your questions via piazza (https://piazza.com/hacettepe.edu.tr/fall2014/bbm101) and you are supposed to be aware of everything discussed in piazza.
- You will submit your work from *https://submit.cs.hacettepe.edu.tr/index.php* with the file hierarchy as below:

```
|-- <student id>
        |------- hw3_part1.c
        |------- hw3_part2.c
```

This file hierarchy must be zipped before submitted (Not .rar , only .zip files are supported by the system).

References

1. http://tutorial.math.lamar.edu/Classes/CalcII/ApproximatingDefIntegrals.aspx