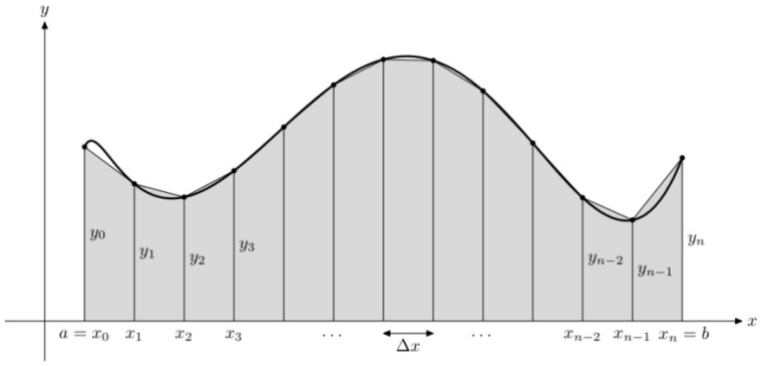
CS 222: Systems Programming

Project #1: Integral Approximation with Functions

Due **Monday of Week 8**

1. Introduction

In this program, you will design functions that allow you to approximate the integral of a function, $f(x)$, using the trapezoidal rule. To find this integral -- the area under the curve -- we can approximate the area as a series of trapezoids, as shown below:



In the figure, the range [a, b] has been divided into *n* different trapezoids, each of which has the same base, $\Delta x = (b - a)/n$. Recall that a trapezoid with base b and sides $h1$ and $h2$ has area:

$$0.5 \times b \times (h1 + h2)$$

Therefore, the area of trapezoid number *k* (1 ≤ *k* ≤ *n*) from the figure above is:

$$0.5 \times \Delta x \times (y_{k-1} + y_k) = 0.5 \times \Delta x \times (f(x_{k-1}) + f(x_k))$$

To find the total area under the curve—and therefore the approximate integral—sum the areas of all trapezoids:

$$\begin{aligned} Area &= 0.5 \times \Delta x \times (y_0 + y_1) + 0.5 \times \Delta x \times (y_1 + y_2) + \cdots + 0.5 \times \Delta x \times (y_{n-1} + y_n) \\ &= 0.5 \times \Delta x \times (y_0 + y_1 + y_1 + y_2 + \cdots + y_{n-1} + y_n) \\ &= 0.5 \times \Delta x \times (y_0 + 2y_1 + 2y_2 + \cdots + 2y_{n-1} + y_n) \\ &= \mathbf{0.5 \times \Delta x \times \left(y_0 + 2 \sum_{k=1}^{n-1} y_k + y_n \right) \approx \int_a^b f(x) dx} \end{aligned}$$

Your integral function will use the equation shown in bold above to approximate the integral, given the endpoints of the interval [a, b] and the number of trapezoids, n.

2. Deliverables

This assignment uses multiple files; starter versions of each file are provided on Canvas:

- ***proj1_integral.c:*** Source file containing your main function.
- ***proj1_functions.h:*** Header file containing function prototypes. **Do not change existing contents of this file**.
- ***proj1_functions.c:*** Source file containing other user-defined functions
- ***Makefile:*** The Makefile to compile your project.
- ***Documented meeting agenda & minutes for at least 2 group meetings.*** Contents must be included: overall flowchart design, work distribution, and testing strategies. Meeting agenda and minutes template is given on Canvas → Modules → Course Intro.

Students must form a team of two and register their team on Canvas under "People → Project 1". Team leaders need to zip all files into an archive named as **proj1.zip**, and submit the zipped archive by uploading it to Canvas. Ensure your file names match the names specified above. Failure to meet this specification will reduce your grade by 10 points.

3. Specifications

Program structure: Your program should follow the general outline below:

1) Prompt for and read the following values:
   - The low and high points of the interval $[a, b]$, over which $f(x)$ is to be integrated.
   - The number of trapezoids, n, to be used in that integration.

If an input error occurs, print an error message and repeat the prompt only for that input. Input errors are as follows:

- `scanf()` cannot read the input values (for example, if the user types "`A 3`" for the interval endpoints).
  - In this case, you must clear the rest of the line (using `badInput()` function) before retrying `scanf()` for the next interation.
  - See "PE2" for a reminder of how to check that the input is properly formatted and how to clear the rest of the line if it is not.
- The "low" interval endpoint is greater than or equal to the "high" endpoint.
- The number of trapezoids is less than 1.

2) Once the user has entered error-free input values, call the `integrate()` function (described later), which will use the trapezoidal method to approximate the integral of $f(x)$ over the interval $[a, b]$ using n trapezoids.
   - Must use integer number comparison for the loop condition – using double type may cause loss of number of iterations. For example, the following is incorrect:

     ```
     double amt;
     for (amt = 0.0; amt < 1.5; amt = amt + 0.0001) { … }
     ```

3) After printing the results, ask the user if he or she wants to repeat the program, and then read a single character for the response to that question. If the user enters:
   - `'Y'` or `'y'` → Return to Step 1).
   - `'N'` or `'n'` → End the program.
   - Any other character → Print an error message and repeat the question.

Other than the aforementioned errors, no additional errors need to be checked. See Section 4 for test cases that demonstrate the proper format for input and output.

Functions: Your program should contain functions with the prototypes shown below:

- `double f(double x);`

  The function being integrated, which should calculate the value:

$$f(x) = \cos(x^2) + \frac{x^3 - 2x^2}{10}$$

  Note that you should include the math library `<math.h>` in order to use the cos() function; this function takes a single argument and returns the cosine of that value.

- `double integrate (double a, double b, int n);`

  This function should use the trapezoidal method to approximate the integral of $f(x)$ over the interval $[a, b]$ using n trapezoids, as described above. The return value of the function is the result of the approximation. Note that the function should not print any values to the screen—the output should be handled in the main function.

- `void badInput();`

  Clear the current line of input. This function should be used after input formatting errors occur.

4. Test Cases

Your output should match these test cases exactly for the given input values. I will use these test cases in grading of your code, but will also generate additional cases that will not be publicly available. Note that these test cases do not cover all possible program outcomes. You should create your own tests to help debug your code and ensure proper operation for all possible inputs. The users' input is underlined and color-coded in green.

```
Enter endpoints of interval to be integrated (low hi): -2.1 1.5
Enter number of trapezoids to be used: 15
Using 15 trapezoids, integral between -2.100000 and 1.500000 is 0.108373

Evaluate another interval (Y/N)? Y
Enter endpoints of interval to be integrated (low hi): -2.1 1.5
Enter number of trapezoids to be used: 45
Using 45 trapezoids, integral between -2.100000 and 1.500000 is 0.109993

Evaluate another interval (Y/N)? y
Enter endpoints of interval to be integrated (low hi): A 3
Error: Improperly formatted input

Enter endpoints of interval to be integrated (low hi): 5 4
Error: low must be < hi

Enter endpoints of interval to be integrated (low hi): 0 0.1
Enter number of trapezoids to be used: W
Error: Improperly formatted input

Enter number of trapezoids to be used: 0
Error: num must be >= 1

Enter number of trapezoids to be used: 10
Using 10 trapezoids, integral between 0.000000 and 0.100000 is 0.099935

Evaluate another interval (Y/N)? Q
Error: must enter Y or N

Evaluate another interval (Y/N)? x
Error: must enter Y or N

Evaluate another interval (Y/N)? N
```

5. Deductions

Deduction Code	Description	Points Deducted
1	Program does not compile, but some code was given.	-80
2	Comments	-10
2a	No header comments	-4
2b	Body of program contains no single line of comment	-4
2c	Other comments such as comments for key variables. Header comments is present but missing some key components	-2
3	File names are incorrect (including source files, Makefile and zip file)	-10
4	Inconsistent program style • Indentation, braces, descriptive variable names, etc. • -2 for each occurrence and maxed at -10	-10
5	Input error	-25
5a	Program must take input as the way given in the sample output. E.g., user types no extra characters other than the numbers.	-5
5b	There are 4 types of input errors to be checked. Deduct 5 points for each missing error type until maxed.	-20
6	Output error	-30
6a	Incorrect integral value	-10
6b	When input error occurs, program does not repeat the correct prompts (incorrect loop structures). Deduct 5 point for each occurrence until maxed.	-15
6c	Printed information does not match with the given sample output exactly. -1 for each occurrence until maxed. • Number of decimal places • Extra newlines in between messages • Not matched prompts and error messages	-5
7	Lack of a Makefile or Makefile cannot compile the code	-5
7a	Makefile does not have a recipe for cleaning the object files and executable or the recipe is incorrect	-2
7b	Makefile does not separate the compilation and linking steps	-3
8	The function prototypes have been modified. Or not all functions in the given start file have been used.	-5
9	Did not work in a team or lack of contributions to the project.	-5
10	Missing project meeting agenda and minutes	-20