Exported for Brian O'Connell on Tue, 27 May 2025 19:34:21 GMT

Programming 10 - Homework

Part 1: Ticker Tape Calculator

The purpose of this program is to showcase how a switch case works. This is something you've seen an example of but we haven't implemented it in any of the languages we've worked in. It will also be a more complex implementation of a file stream in C++.

If you've never seen a ticker tape calculator, it's a calculator that keeps a record of all the inputted arguments through a scrolling tape. These were very popular among accountants (my father loved his) in the late 80s and 90s.

If you want to see one working, check out this ASMR video with one: https://www.youtube.com/watch?v=CRbtoGJmnNA

We're going to create a version of this in c++ that produces its own ticker tape feed in the form of an output file.

Part 1 Walkthrough

Create a new C++ Ticker Tape Calculator that

- Creates a 'ticker tape' file for storing calculation info to a specified accuracy
- Handles several operators to perform actions

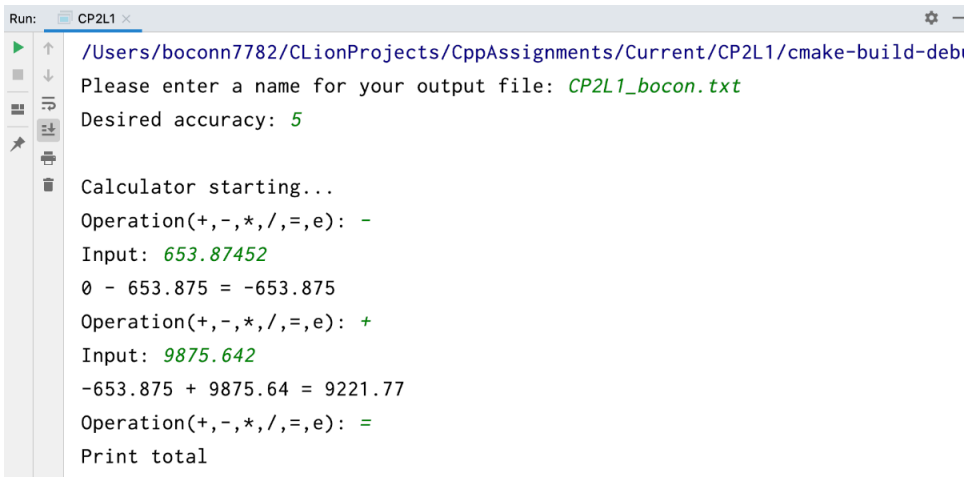
    - `=` - Prints the total to the output file and resets it to 0
    - `e` - Ends the entire program
    - `+,-,*,/` - performs the operation to an input argument and running total
- Prints the individual calculations to the console window
- Keeps a record of the input argument and mathematical operator in the output file

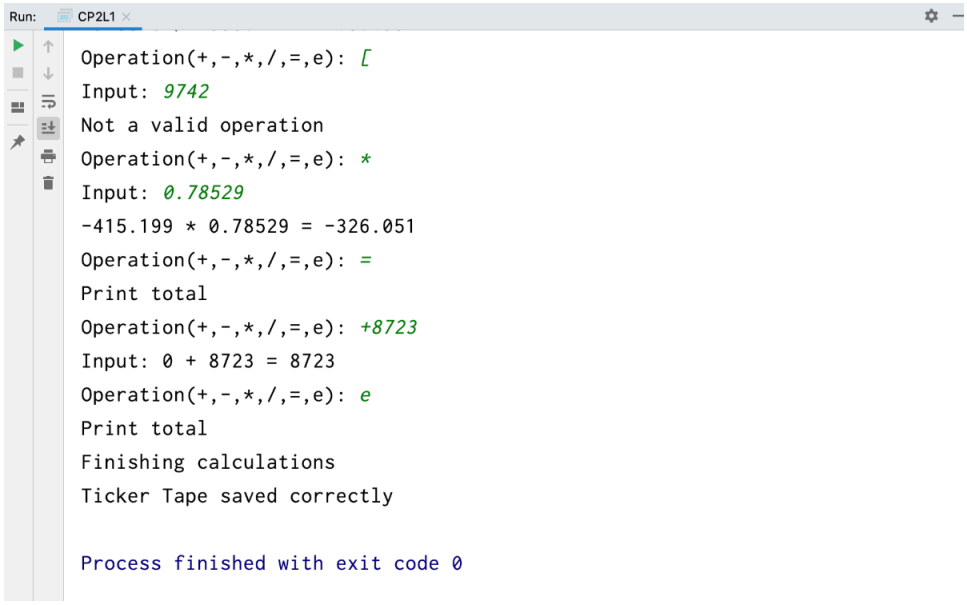
Requirements:

- Use a user input for the filename and accuracy for the
- Use if else if for handling the `=`, `e` or the mathematical operators as a category
- Use a switch case for handling the mathematical operations
    - Must be able to handle an incorrect character
- To the console window:
    - Display the full equation and answer of each calculation
- To the output file:
    - Keep a record of each mathematical operator and input argument
    - Print the total only when the command is given to

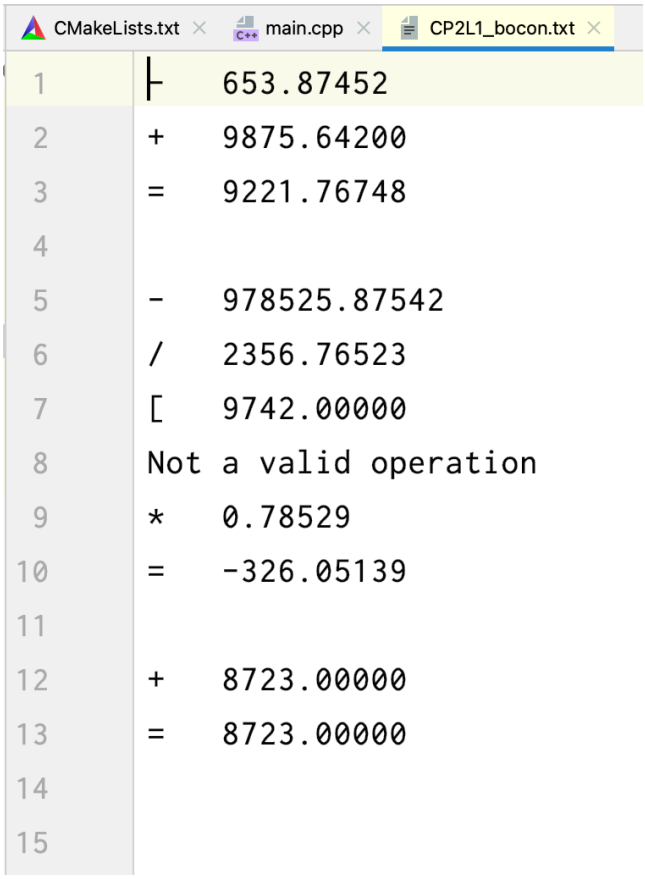
Examples outputs:

Console Window:





Output File:



Pseudocode:

This could be achieved by a variety of approaches, but there are some very specific things I want you to practice so implement the following pseudocode. The output file will be referred to ticker tape:

```
PROGRAM TickerTapeCalculator:
Introduce Program to user
Ask user for filename for ticker tape
Specify accuracy for output
Establish output file stream
WHILE 1
  Ask user for intended operation (+,-,*,/,=,e)
  IF operation is = or e
    Print total to ticker tape
    reset total
    IF operation is e
      end forever loop
    ENDIF
  ELSE
    ask user for numerical input
    Print operation and numerical input to ticker tape
    SWITCH operation
      CASE +,-,*,/
        Calculate update to total based on operator and input
        Print mathematical operation and updated total to console
      CASE default
        Print warning to console and ticker tape
    ENDSWITCHCASE
  ENDIF
ENDWHILE
Check the output file was created
ENDPROGRAM
```

Here's a link to some starter code with the pseudocode as comments:

https://raw.githubusercontent.com/boconn7782/CourseCode/refs/heads/main/CPP/P10H1_Starter

Build your program

Start a new project

Start a new project using CLION. User the pseudocode above as the start to your comments and to help scaffold your program. Don't forget to include your program

Set up the parts of your program

1. Libraries & Directives

We have requirements that will need certain libraries:

- Using strings
- Communicating with the console window
- Creating an output file stream and communicating with it
- Format requirements for those communications

2. Declare Variables

We already know of some information we will need to store and use throughout:

- An output stream object will be needed for
- The user must establish the filename
- The user must define the accuracy of information recorded to the output
- The user must input an operation based on a limited set of characters
- The user must input a numerical value to be used in a calculation
- A total must be updated by each calculation

3. Main Program

- We know what we want to do in our main program based on the pseudocode

Integrate this into your code:

Build the code from the pseudocode

Title Block:

- `/* Title Block */`

The Title Block and the introduction & instructions can typically wait till the end to be fully completed but a first pass will at least make it easier to remember what this is about if you have to stop and come back to it. It's partially completed in the starter code so you can come back and clear it up later.

Libraries & Directives

- `#include... //Load the necessary libraries`

Based on what has been established as required, load the necessary libraries. A good guess would be at least the ones introduced in this lesson.

- `using namespace std;`

Requirement: Output File Stream

- `int main()`
- `{`
- `  // Introduce and Instructions for user`
- `  // Ask user for filename for ticker tape file`
- `  // Establish output file stream`

The first major element you know you need to implement is an output file stream. Create an output file stream object and connect a file using the file name defined by the user. When using a string as an input argument for `ofstream` commands, you need to remember to convert it by appending `.c_str()` to the variable.

Requirement: output accuracy

- `  // Ask user for accuracy output`

When setting formats for a stream, you only need to do it once, so might as well get it done here before starting to write to the output file.

Requirement: Ending operations

- `  while(1) {`
- `    // Ask user for intended operation (+,-,*,/,=,e)`
- `    IF operation is = or e`
- `      Print total to ticker tape`
- `      reset total`
- `      IF operation is e`
- `        end forever loop`
- `      ENDIF`
- `    ELSE`

Your logic doesn't always have to be in the order of typical operations. Here we have the first `if()` handling the end condition, entering '`e`'. So the gist of this is when '`e`' or '`=`' are entered we want the total to get printed to our ticker tape file and then reset the total. In addition, for '`e`' we want to end the forever loop.

We use the first if to handle what we want both to do and a nested if statement for the special case of '`e`'. If you look below, there are things to be done outside of the forever loop. This criteria should only end the loop, not the entire program.

Requirement: Calculation Inputs

- `ENDIF`
- `ELSE`
- `ask user for numerical input`
- `Print operation and numerical input to ticker tape`

We want the user to enter their operation and the next operand. We also want to capture that on the ticker tape to have a history of operations. Write that info to the output file. Use the example image above to help with formatting.

Requirement: Switch Case for Calculations

- `SWITCH operation`
- `CASE +,-,*,/` <-- This means a separate case for each of those
- `Calculate update to total based on operator and input`
- `Print mathematical operation and updated total to console`
- `CASE default`
- `Print warning to console and ticker tape`
- `ENDSWITCHCASE`

For help with implementing the case statements, look to the lesson content. For output formats, look to the example outputs provided above. Don't forget about syntax and need for `break` in case statements. The example output formats are fairly straightforward, don't try and overly analyze spacing or anything, just print the correct information to the correct places.

Requirement: Check you files

- `ENDIF`
- `ENDWHILE`
- `Check the output file was created`
- `ENDPROGRAM`

For this you will have to close your output file stream to the file and then open an input file stream object connected to that file. If that works and creates a valid object, then your original file saved correctly.

Part 2: Baby Name Counter

The Social Security Administration publishes lists of the most popular names given to babies born each year on their website. Here is a list of the most popular names since 1915 - https://raw.githubusercontent.com/boconn7782/CourseCode/main/CPP/BabyNames.txt

The first column is the year, the second is the most popular girl's name for that year, and the third column is the most popular boy's name. Each new row is a new year, starting at 1915 and continuing until the most recently available data. You will use this for your assignment so save a copy of this or create a new file called *BabyNames.txt* and copy the content into it.

Write a C++ Program that:

- Welcomes the user
- Introduces and explains the application's purpose to the user
- Describes the program's purpose
- Asks the user for the input file name
- Tests if the file opens correctly
- Reads the data into a 2D array in the main program
- Asks the user for a name
- Either:
    - Returns, in a full sentence(s), the first year in which that name was most popular and how many times it has been the most popular name.
    - Returns a statement telling the user that the name has not been one of the most popular names in the last 100 years.
- Asks the user if they want to do another search until they respond "n"

Requirements

- Comment your code!!
- Test your code using
  - A least 1 Male and 1 Female name that were on the list
  - At least 4 different names
    - 1 must be your name
    - 1 must not be in the list (This could be your name)

Assignment Summary

Part 1: Ticker Tape Calculator

- Creates a 'ticker tape' file for storing calculation info to a specified accuracy
- Handles several operators to perform actions
  - `=` - Prints the total to the output file and resets it to 0
  - `e` - Ends the entire program
  - `+,-,*,/` - performs the operation to an input argument and running total
- Prints the individual calculations to the console input
- Keeps a record of the input argument and mathematical operator in the output file

Part 2: Baby Name Counter

- Loads a list of popular baby names by year and gender into an array
- Asks the user for a name
- Either:
  - Returns, in a full sentence(s), the first year in which that name was most popular and how many times it has been the most popular name.
  - Returns a statement telling the user that the name has not been one of the most popular names in the last 100 years.
- Repeats until user responds “no”

Submission Requirements:

Part 1: Ticker Tape Calculator

1. Source Code - .cpp file
2. Command Window Output - .pdf file

    1. Reminder, your output should include an example or examples using a combination of a negative and positive #s as well as integers and non-integers.
    2. You must also showcase a response to an incorrect input
3. Output file - .txt file

Part 2: Baby Name Counter

1. Pseudo Code/ Flowchart - .pdf
2. C++ code - .cpp
3. Console output - .pdf
    1. Run at least 4 times - Using your first name, a name not on the list, 1 male and 1 female you know is on the list.