COMP 354 , Summer 2021
Project

Team C

August 9, 2021

Table 1: Team C

Name
Sammy Chaouki
Mingyang Chen
Alessandro Ciotola
Ruslan Dallin
Diouma Dembele
Jiayue Deng
Xin Yi Deng

Contents

Introduction

1.1 Overview

The goal of our project is to build an elementary scientific software, called ETERNITY. ETERNITY is a scientific calculator which has the capability of parsing and evaluating mathematical expressions, where a mathematical expression is composed of one or more functions which will be manually implemented by our team members.

Before building ETERNITY, we conducted several interviews to gather information and calculator usage of potential users. Based on the data collected, we created seven personas which represent seven types of users. Finally, we build use cases to model potential ways of using ETERNITY.

1.2 Collaboration Patterns

Collaboration patterns help create an awareness of possible problems that can occur in a group project and how to overcome them.
The following is an initial list of collaboration patterns adopted by our team. It is taken from the study ”Improving Student Group Work with Collaboration Patterns: A Case Study” [1].

1. **Clear up questions** - Ensure that there are no open questions when starting with the project. At the end of each meeting, dedicate some time to answer questions.

2. **Share expectation** - Know what team members expect to get out of the project and what effort they will put into it.

3. **Give a first warning** - React on malfunctioning group members in a constructive but clear way.

4. **Fill knowledge gaps** - If team members are missing knowledge necessary for fulfilling their tasks, then support them with acquiring this knowledge.

5. **Centralize Work Product Management** - Ensure that all team members always have access to the latest project artefact versions.

6. **Manage the project** - Define appropriate roles, tasks, and responsibilities for managing the project.

7. **Mediate the dispute** - Don't let disputes become real problems, but mediate them as soon and as constructive as possible.

8. **Keep motivated** - If motivation to work on the project goes down, change tasks and work actively on being motivated again for the rest of the project.

9. **Start Immediately** - Don't lose valuable time by postponing to start working on the project.

10. **Regularly check requirements fulfillment** - Check regularly if the (intermediate) artefacts meet the requirements instead of waiting until the end, as the artefacts likely cannot be corrected by then.

11. **Spread tasks appropriately** - Ensure that all team members have an appropriate amount of tasks, according to their skills and responsibilities.

Tools used to support collaboration patterns

- **Facebook Messenger** - Messenger's voice calling feature is used for group meeting. It allows us to talk about the project, assign tasks, know where everyone's progress and solve issues. Messenger's chat feature is used to set up group meetings, answer questions and post announcements.

- **Google Drive and Google Doc** - All artifacts of the project can be found in the Google Drive. This ensures that all team members have access to latest project artifact. We also have our meeting summaries which lists everyone's tasks and the deadlines to ensure everyone is on the same page.

- **Github** - Github is used for version control and source code management. It allows us to share our code and help each other when necessary. We also review each other's code before merging it to the main branch to ensure all codes work as it should. We will use Git standards to avoid potential problems of version control.

- **Overleaf** - Overleaf is used to produce our documentation in LaTex. It has a feature which allows us to collaborate simultaneously on the same document.

1.3 Functions: Definition and Allocation

Definitions

- $\arccos(x)$ - The arccos function is an inverse trigonometric function. It is the inverse of the cosine function (cos(x)). It returns the angle whose cosine is a given number. The real variable x must be between -1 and 1 [2].

- ab^x - An exponential function is a function of the form $f(x) = ab^x$, where a and b are real constants, and the real variable x occurs as an exponent [3].

- $log_b(x)$ - The logarithmic function of the form $y = log_b(x)$, where b is a real positive constant and x is a real variable, is the inverse of an exponential function, $b^y = x$, so that the independent variable appears in a logarithm. [4]

- $\Gamma(x)$ - The gamma function is defined to be an extension of the factorial to complex and real number arguments. It is related to the factorial by $\Gamma(x) = (x - 1)!$, where x is a real positive variable. [5]

- MAD - The mean absolute deviation (MAD) of a data set is the average distance between each data point and the mean. It gives us an idea about the variability in a data set. [6]

- SD - The standard deviation is a statistic that measures the dispersion of a data set relative to its mean. The standard deviation is calculated as the square root of variance by determining each data point's deviation relative to the mean. [7]

- sinh(x) - The hyperbolic sine function is a hyperbolic function whose value of the complex number x is one-half the difference between the exponential of x and the exponential of $-x$, such that $\sinh(x) = \frac{e^x - e^{-x}}{2}$ [8].
- x^y - An exponential function in the the form $f(x) = x^y$, where x and y are both real variable.

Function Allocation

- $arccos(x)$ - Alessandro Ciotola
- ab^x - Diouma Dembele
- $log_b(x)$ - Mingyang Chen
- $sinh(x)$ - Sammy Chaouki
- MAD - Xin Yi Deng
- SD - Jiayue Deng
- x^y - Ruslan Dallin

Interviews

The interviews we have conducted were to give us a better understanding of our potential users and how we should design and implement our software. As a team, we decide to follow the funnel methodology, meaning we start with more general questions and slowly move towards more specific ones.

2.1 Questions

Here is the list of questions that were asked during our interviews:

- **General Questions**
  1. How old are you?
  2. What is your educational background?
  3. Are you currently a student or employed?
     - If you are working, in which domain/field are you currently working in, and does it involve mathematics?
  4. Tell us a little about yourself.
  5. On a scale of 1 to 10, how comfortable are you with using a computer (1 been not comfortable at all, 10 been very comfortable)
  6. Have you ever heard or used something called the command line/ command prompt?
     - If yes, how often have you used it and for what cases?
     - Do you prefer using the command line/prompt or a GUI (graphical user interface)?
  7. Which operating systems are you currently using?

    - Are you satisfied with the calculator on your system? Why or why not?

8. Do you have a number pad on your computer?

- **Specific Questions**

    1. How often do you use a scientific calculator versus a standard calculator in your daily life?
    2. How likely are you to use a computer's scientific calculator rather than a portable one?
        - On which device do you most often do your calculations (computer, online calculator, phone, a physical one, ipad, etc.)?
    3. When you use a calculator on your computer, do you type with your keyboard, click on the numbers with your mouse or use a touchscreen?
        - Which method do you prefer?
    4. If you are often using calculators in your work, what are the most common functions or constants that you use?
        - Do you use many advanced functions on a calculator? If yes, which ones?
        - Which functions of the following have you used in the past?
        - Which ones do you find the most useful?
            - List of functions from the assignment:
                - $arccos(x)$
                - ab^x
                - $log_b(x)$
                - $\Gamma(x)$
                - MAD
                - σ
                - x^y
    5. What are the most important features you look for in a scientific calculator? (e.g.: memory keys, history of calculations (ans), correctly displaying the functions, conversion between decimals and fractions, arrow keys, menu screen, converting to binary)

6. Could you suggest a feature that could be implemented that would help improve the experience of using a calculator on a computer? Explain.
7. What features on the calculator do you not use? Why?
8. Do you look for simplicity in a calculator (i.e. expand the calculator functionalities only if needed)
9. If you have any problems or dissatisfaction's with current calculators, what are they? Why?
10. Is the design of the calculator more important than the functionality of it? Why?
11. Do you think that a help section (legend) for the calculator should be included in the program (to help the user understand the functionalities)?
12. Which online calculator do you use while solving mathematical equations, and what is the design and available inputs? (e.g. symbolab, WolframAlpha. . .)
13. Do you use any professional math software (e.g. R-studio, maple, mathematica)?

- **Closing questions**
  1. Are there any comments that you would like to add?
  2. Do you have any questions for us?

2.2 Responses

For the responses we decided to give a summary of each question based on the 9 interviews we have conducted. See table on page 12 for the responses (the responses are matched with the questions mentioned in section 2.1)

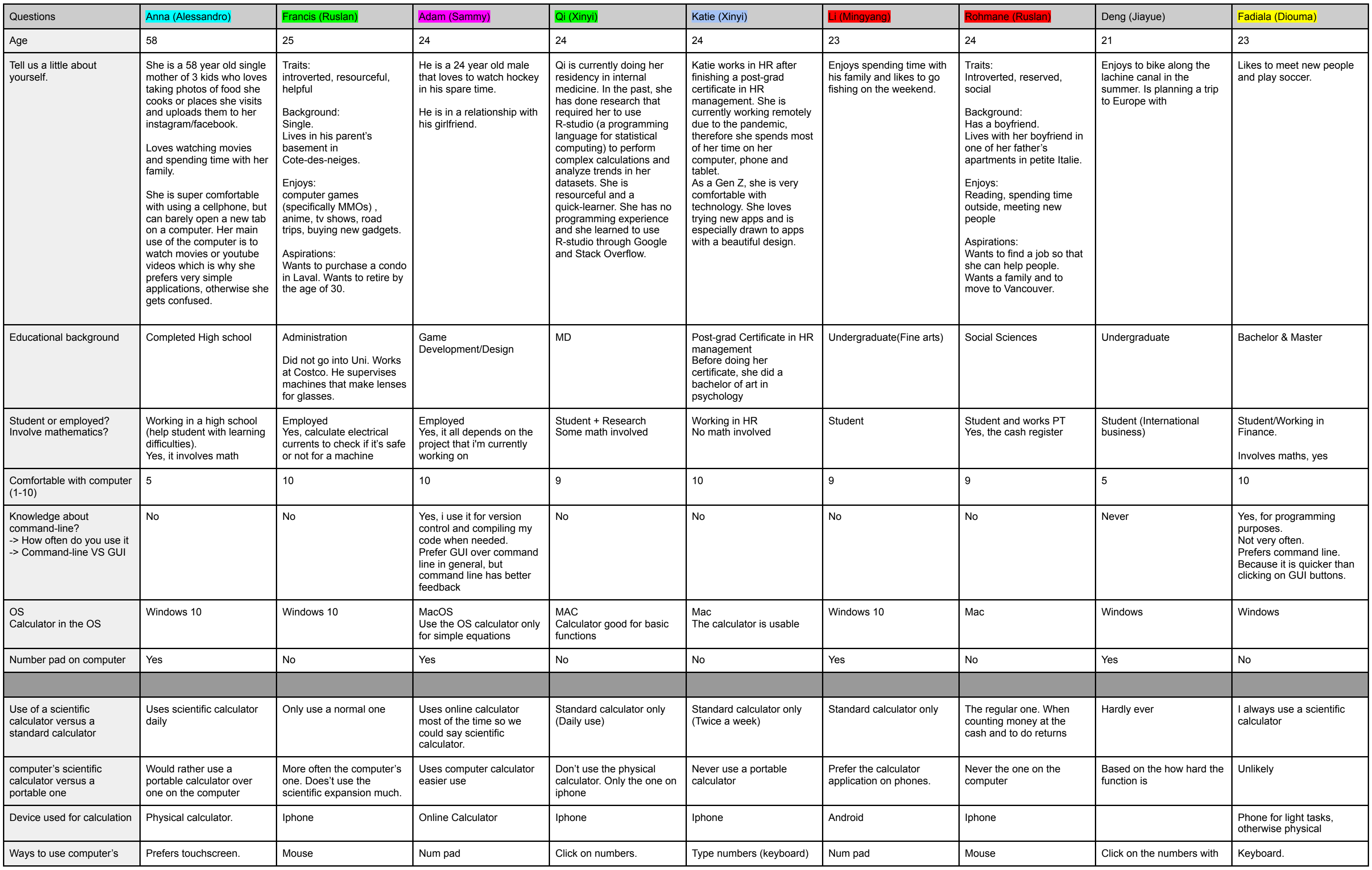
- **General Questions**
  1. The age group covered during our interviews varied between 21 to 58
  2. In general, the average interviewee has completed a undergraduate diploma or is currently pursuing higher education.

3. We noticed that most of our sample size are currently working and that on average their jobs involve some mathematics.
4. As stated previously, we have a diverse data set going from 21 year old students to a 58 year old teacher.
5. In average, our interviewees were pretty comfortable with their computer (about 8.5 out of 10).
6. Our sample size gives us a good understanding that in general people are not comfortable using a command line and in general prefer a GUI interface as it is easier to use.
7. The operating system used vary between Mac OS and Windows.
   - The satisfaction of our interviewees with their native calculator varies, some are more than satisfied while others are more comfortable using an electronic ones (mainly for ease of use and functionality).
8. Up to 50% of them did not have access to a numeric keypad on their computer.

- **Specific Questions**

  1. Our data set provides a good statement on who uses a scientific calculator, as it all depends on their work/education. People that have math involved in their field use scientific calculators regularly while the other part of the sample size are content with a basic arithmetic calculator.
  2. The average interviewee resort to a online calculator when it comes to computing mathematical equations either on their phone, on a website or the operating system calculator.
  3. More than 50% of them are currently comfortable using a mouse to enter mathematical equations on their online calculator.
  4. All of our interviewees are currently using all the basic arithmetic operators, ranging from addition to division, as for the constants we noticed that π is the one that was commonly used. For more advanced functions, we noticed that most interviewees use the basic trigonometry identities (sin,cos,tan). From the list of functions that were given for the assignment, only $log_b(x)$, ab^x and x^y have been deemed as useful.

5. On average, our data set shows that having a conversion of decimal to fractions, history of calculations and memory keys is a must for any online or electronic calculator.
6. As stated in the previous point, it is essential to have access to a history of calculations that were made, decimal to fraction, as well as a easy to use interface such that the calculator is not limited to a certain group of users.
7. From a general perspective, most of the interviewees have stated that most function that are not basic arithmetic are seen as not useful features.
8. Our interviewees have stated that simplicity is something that they always look for when using a calculator either physical or online
9. In general, our sample size are satisfied with their current calculator as it provides all the functionalities they are looking for.
10. Over 90% of our data set have agreed that functionality is more important than design as a calculator has a single use, which is to compute mathematical equations.
11. The interviewees have been vocal on having a good help section to provide a ease of use for new users that are unfamiliar with new functionalities or where to find them.
12. Only 22% of our interviewees are currently using online calculators, primarily Symbolab
13. Most of our sample size are not using professional software calculators to compute mathematical equations.



Questions	Anna (Alessandro)	Francis (Ruslan)	Adam (Sammy)	Qi (Xinyi)	Katie (Xinyi)	Li (Mingyang)	Rohmane (Ruslan)	Deng (Jiayue)	Fadiala (Diouma)
Age	58	25	24	24	24	23	24	21	23
Tell us a little about yourself.	She is a 58 year old single mother of 3 kids who loves taking photos of food she cooks or places she visits and uploads them to her instagram/facebook. Loves watching movies and spending time with her family. She is super comfortable with using a cellphone, but can barely open a new tab on a computer. Her main use of the computer is to watch movies or youtube videos which is why she prefers very simple applications, otherwise she gets confused.	Traits: introverted, resourceful, helpful Background: Single. Lives in his parent's basement in Cote-des-neiges. Enjoys: computer games (specifically MMOs) , anime, tv shows, road trips, buying new gadgets. Aspirations: Wants to purchase a condo in Laval. Wants to retire by the age of 30.	He is a 24 year old male that loves to watch hockey in his spare time. He is in a relationship with his girlfriend.	Qi is currently doing her residency in internal medicine. In the past, she has done research that required her to use R-studio (a programming language for statistical computing) to perform complex calculations and analyze trends in her datasets. She is resourceful and a quick-learner. She has no programming experience and she learned to use R-studio through Google and Stack Overflow.	Katie works in HR after finishing a post-grad certificate in HR management. She is currently working remotely due to the pandemic, therefore she spends most of her time on her computer, phone and tablet. As a Gen Z, she is very comfortable with technology. She loves trying new apps and is especially drawn to apps with a beautiful design.	Enjoys spending time with his family and likes to go fishing on the weekend.	Traits: Introverted, reserved, social Background: Has a boyfriend. Lives with her boyfriend in one of her father's apartments in petite Italie. Enjoys: Reading, spending time outside, meeting new people Aspirations: Wants to find a job so that she can help people. Wants a family and to move to Vancouver.	Enjoys to bike along the lachine canal in the summer. Is planning a trip to Europe with	Likes to meet new people and play soccer.
Educational background	Completed High school	Administration Did not go into Uni. Works at Costco. He supervises machines that make lenses for glasses.	Game Development/Design	MD	Post-grad Certificate in HR management Before doing her certificate, she did a bachelor of art in psychology	Undergraduate(Fine arts)	Social Sciences	Undergraduate	Bachelor & Master
Student or employed? Involve mathematics?	Working in a high school (help student with learning difficulties). Yes, it involves math	Employed Yes, calculate electrical currents to check if it's safe or not for a machine	Employed Yes, it all depends on the project that i'm currently working on	Student + Research Some math involved	Working in HR No math involved	Student	Student and works PT Yes, the cash register	Student (International business)	Student/Working in Finance. Involves maths, yes
Comfortable with computer (1-10)	5	10	10	9	10	9	9	5	10
Knowledge about command-line? -> How often do you use it -> Command-line VS GUI	No	No	Yes, i use it for version control and compiling my code when needed. Prefer GUI over command line in general, but command line has better feedback	No	No	No	No	Never	Yes, for programming purposes. Not very often. Prefers command line. Because it is quicker than clicking on GUI buttons.
OS Calculator in the OS	Windows 10	Windows 10	MacOS Use the OS calculator only for simple equations	MAC Calculator good for basic functions	Mac The calculator is usable	Windows 10	Mac	Windows	Windows
Number pad on computer	Yes	No	Yes	No	No	Yes	No	Yes	No
Use of a scientific calculator versus a standard calculator	Uses scientific calculator daily	Only use a normal one	Uses online calculator most of the time so we could say scientific calculator.	Standard calculator only (Daily use)	Standard calculator only (Twice a week)	Standard calculator only	The regular one. When counting money at the cash and to do returns	Hardly ever	I always use a scientific calculator
computer's scientific calculator versus a portable one	Would rather use a portable calculator over one on the computer	More often the computer's one. Does't use the scientific expansion much.	Uses computer calculator easier use	Don't use the physical calculator. Only the one on iphone	Never use a portable calculator	Prefer the calculator application on phones.	Never the one on the computer	Based on the how hard the function is	Unlikely
Device used for calculation	Physical calculator.	Iphone	Online Calculator	Iphone	Iphone	Android	Iphone		Phone for light tasks, otherwise physical
Ways to use computer's	Prefers touchscreen.	Mouse	Num pad	Click on numbers.	Type numbers (keyboard)	Num pad	Mouse	Click on the numbers with	Keyboard.

calculator (type, click, touch)								mouse	
Most common function/constant use in work.	Uses standard inputs (+/*-) along	Multiplication	Basic operators, and log base 10	Basic arithmetic (add, subtract, multiply, division)	Basic arithmetic (add, subtract, multiply, division)	Basic arithmetic	Multiplication	Square root, sin and cos	Addition, multiplication, basics
Do you use many advanced functions on a calculator? If yes, which ones?	%, pi, sqrt and x^y	No	Yes, Summation function and others.	No	No	No	No		Yes, sqrt, exponents, logs
From the list of function (Used previously, most useful)	ab^x, x^y Doesn't know the rest.	Doesn't use the advanced ones however, after asking him which ones he knows: Xy	logb(x) → most useful also	logb(x), SD (most useful) x^y	SD	x^y	arccos(x), Have used. Not useful Abx, Yes, useful logb(x), Γ(x) (Gamma Function), MAD (Mean Absolute Deviation), σ (Standard Deviation), sinh(x), xy Yes, useful		In the past: log, standard deviation. Most useful from assignment: log
Rank important feature in the calculator (memory keys, history of calculations (ans), correctly displaying the functions, conversion between decimals and fractions, arrow keys, menu screen, converting to binary)	Conversion of decimal to fractions.	Arrow keys	Good instructions on how to properly use the software, History of calculations to be able to look back to previous executions	Memory: 9 History of calculation: 5 Displaying the functions: 5 Conversion between decimals: 4 Arrow keys: 2 Menu screen: 2 Conversion to binary number: 1	Memory: 10 History of calculation: 9 Displaying the functions: 10 Conversion between decimals: 10 Arrow keys: 6 Menu screen: 2 Conversion to binary number: 1	History of calculation: 10 Memory: 9 Displaying the functions: 7 Conversion between decimals: 6 Arrow keys: 7 Menu screen: 3 Conversion to binary number: 0	Correctly displaying the functions (the natural function view)	Be able to calculate the functions	Correctly displaying fractions and long computations.
Suggest a feature for calculator	Keeping history of calculations.	spreadsheet calculations like excel	Clear and easy help manual since most online scientific calculators dont have those.	None	Showing all the inputs and output on the same screen E.g.: On the iphone, when you do 2 + 3 = 5, you only see one number at a time and you can't check your previous input	The feature that can convert decimals into fraction such as: 0.2 -> 1/5	little games for when you're bored. LIke the chrome dinosaur	Nice user interface	Not really, I don't use it very often.
Feature not used	Only simple features, no sin,arcsin, etc.	He only uses the simple features	Everything related to stats	Percentage, +/- (iphone calculator)	Only uses the basic function on iphone calculator	Only basic arithmetic and x^y	Doesn't use many features.	Many operating functions that are not helpful for my field.	Basic computations, for studies, work, life in general
Simplicity in calculator	Yes	YES!	Yes	Yes	Yes Expansion only for parenthesis	Yes	Yes, very!	No	Not really.
Problems or dissatisfaction with current calculator	Doesn't like having many buttons that are useless to her (prefer simple)	Does Not have any	Satisfied	None	Cannot see previous input	None	She would like to see explanations of each function (what they do)	None	No
Design vs Functionality	Functionality. "No point in having a nice design if it doesn't work properly."	Functionality	Design over functionality, easier design makes it better for user experience	Functionality	No But as a visual person, nice to have a nice design	Functionality	No, she doesn't care about the design. Functionality more important	Functionality	Design so it's easier to hold in your hands.
Help Section	Yes	Sure	Yes mandatory	No, I only use the functions that I know how it works	No	No	Yes!	Yes	Yes, for those who don't know them
Use of online calculator	None	Never used	Yes	No	No	No	Google calculator	symbolab	I don't really use any.
Professional Calculator	None	No	No	Yes (R-Studio)	No	No	No	No	None

2.3 Analysis of interview responses

We analyzed the similarities and differences in responses from the nine interviews and created seven different archetypes/personas based on them.

Alexandra - Created using Rohmane's and Li's collected data. Alexandra represents an ETERNITY user that uses mainly basic functions, such basic arithmetic (addition, subtraction, multiplication, and division) and a few advanced functions, such as exponential and logarithmic functions. She is an undergraduate in a major that does not use advanced functions. She is comfortable with technology, however, she has limited to no knowledge of the command-line. Thus, she prefers applications with a graphical user interface. A simple calculator with only the basic functions is good enough for her. Although, if the software calculator can expand to a scientific calculator, it would be useful in some of her math courses. The implementation of a small game on the calculator could entertain her when she is bored.

Anthony - Created using Adam's collected data
Anthony represents a user that will utilize most features ETERNITY has to offer. He works in the game development industry, which requires some advanced mathematical knowledge (such as physics equations, etc.). As a software developer, he uses the command-line from time to time and generally prefers a GUI. Although, he admits that the command line has better feedback, he knows what he wants in a software calculator. A clear and easy help manual is essential for him and this is something that is missing in most online calculators. The design is also important for better user experience.

Benoit - Created using Francis' collected data
He represents a user whose usage of ETERNITY will be quite limited. He has a lower education background compared to most of the other interviewees. At his job, he uses basic arithmetic to ensure the functioning of machines that make lenses for glasses. He mainly uses the calculator that is on the phone since he always carries it around him. He does not like the complexity of scientific calculators and never uses the expansion feature of the phone's calculator.

Katherine - Created using Katie's collected data

Katherine represents a user whose usage of ETERNITY will be quite limited. She has a post-graduate certificate in HR management and she is currently working in HR. She only uses basic arithmetic operations in her day-to-day life. She values simplicity in her calculators. However, a good design is always aesthetically pleasing. To differentiate with other similar personas, we added the use of Excel in her daily work. The ability to calculate multiple data points with a few clicks is important for this persona.

Maria - Created using Anna's collected data
Maria represents a user that will utilize most features ETERNITY has to offer. She is a high school educator and teaches mathematics to high school students. She is really familiar with advanced mathematical functions such as exponential functions, logarithmic functions, trigonometric functions, basic statistics, etc. She mainly uses an electronic scientific calculator. She teaches her students how to use these calculators since most of them use it for the first time. She is not proficient with computers, as she mainly uses it to watch movies and YouTube.

Olivier - Proto-persona to represent an advanced user.
Olivier represents a user that will utilize all features ETERNITY has to offer. He works in the financial industry and uses many advanced functions to build complex financial models. He has a strong mathematical and computer background. He has tried many types of software to do his calculations. He uses his phone to perform basic operations and he uses professional statistical software for his work. With Olivier, we want to create a persona that will use all functions in ETERNITY and that has a good knowledge of programming.

Sarah - Created using Qi's collected data
Sarah represents a user that will utilize most features ETERNITY has to offer. She is currently doing her residency in internal medicine and does stem-cell research on the side. She uses a basic calculator for her daily life and to do prescriptions. For her research, she uses R-studio since she needs more advanced functions and she often has thousands of data points to analyze and graph. She has little to no programming experience. When using R-studio, she relied heavily on Google and Stack Overflow to figure out how it works. Therefore, a good and clear help section is crucial for her.

Personas

3.1 Persona 1 - Alexandra

Personal profile

- Female, age of 24
- Studies social sciences as an undergraduate student.
- Works part-time as a cashier in a magazine shop.
- Little Italy, Montreal

- She enjoys reading in her spare time. Although she is an introvert, she likes to go out and meet new people. She would love to find a job in Vancouver that will have a positive impact on people and start her own family.

Usage of calculators and functions

- As a cashier, Alexandra relies heavily on a basic calculator to perform basic arithmetic operations. She needs to calculate the amount customers need to pay and process refunds. She also uses exponents and trigonometric operations in her math courses.

Usage of software and hardware platforms

- For simple calculations, Alexandra prefers Google's calculator over the default one of her Windows computer.
- For her math class, she has used Symbolab and Wolfram Alpha to help her solve more complex equations.
- She does not use the command line frequently; however, she used it once to fix a problem she was having on her computer. She mentioned that she felt a little lost using it and prefers to use a GUI instead.
- Alexandra wishes she had a number pad on her computer as she has gotten used to typing numbers quickly on the cash register.

Calculator preferences

- Alexandra values simplicity and ease of use over functionality of a calculator.
- She would like to be able to see the history of her previous calculations and to be able to play littles games like Google Chrome's dinosaur game when she is bored.

Influencers that surround the persona that may influence choices

- Her classmates, teachers and colleagues may influence her. And if there is any better choice of calculator that is suggested by those influencers, she would like to give it a try.

3.2 Persona 2 - Anthony

Personal Profile

- Male, age of 24
- Studies completed: undergraduate game design
- Works full-time in a game development company
- Lives in NDG, Montreal
- He is very sociable and he likes going to watch Hockey with his friends. He is a big supporter of the “Montreal Canadiens”. He is currently thinking about whether he should continue with a masters degree or take some years off to travel around the world as he likes to discover new cultures.

Usage of calculators and functions

- His job requires him to perform some physics equations and some calculus when working with video games.
- He uses basic arithmetic operations, root functions, logarithms and exponents on a daily basis.

Usage of software and hardware platforms

- He uses a Windows computer at all times as he despises MacOS and uses the default calculator for most of his calculations at home.
- When he is at work, he uses his TI-84, a graphing calculator, that he has kept since high school. He heavily relies on Wolfram Alpha to double check some of his calculations.
- Anthony uses the command line very frequently. He learned how to use it during his undergraduate studies and he prefers using it over the GUI to maximize his productivity.
- He is a big fan of his number pad, it helps him type numbers very quickly.

Calculator preferences

- He values functionality over simplicity. Having studied game design, Anthony got used to using very complex software and he prefers being able to utilize every function a program has to offer if it can maximize his productivity.
- He needs a calculator that can map certain functions to some custom keys and would like to be able to convert between decimals and fractions easily. He also mentioned that a menu screen would be a nice feature to have as it may allow for more customization which he is a big fan of.

Influencers that surround the persona that may influence choices

- Some of his coworkers use professional math software to calculate some expressions. Therefore, they are trying to influence him to change.

3.3 Persona 3 - Benoit

Private information

- Males, age of 25
- Studies completed: CEGEP
- Works full-time at Costco overseeing the lens making machinery
- Côte-des-Neiges, Montréal
- He currently lives in his parent's basement. He enjoys computer games (specifically MMOs), anime, tv shows, road trips, and buying new gadgets. He would like to start investing in real estate and the first step for him would be buying a condo in Laval so that he can retire by the age of 30. His friends qualify him as introverted, helpful and resourceful.

Usage of calculators and functions

- His job rarely requires the use of a calculator and even less of a scientific calculator. He was only asked on one occasion to calculate the threshold of an electrical current generated from a machine using a formula that made use of multiplications and divisions.

Usage of software and hardware platforms

- Benoit prefers using the calculator on his Iphone for all his calculations as it is easy to use and always close to him.
- He does not know what the command line is and thus has never used it.
- Benoit has a full sized keyboard but rarely uses the number pad.

Calculator preferences

- Benoit demands simplicity over functionality as he gets easily lost when programs cramp too many functions.
- He would like to have the ability to access an easy to use help screen showing how to use each of the functions and features. He would also like the calculator to display functions as naturally as possible (like they are in a textbook).

Influencers that surround the persona that may influence choices

- His friends and coworkers might influence him when it comes to the decision of a calculator as he does not use calculators often and values their input.

3.4 Persona 4 - Katherine

Private information

- Female, age of 24
- Studies completed: Post-grad diploma in human resource management
- Works full-time in human resources as a hiring manager
- Currently living in Toronto
- Currently working remotely due to the pandemic, therefore she spends most of her time on her computer, phone and tablet. As a Gen Z, she is very comfortable with technology. She loves trying new apps and is especially drawn to apps with a beautiful design.

Usage of calculators and functions

- Her work often requires her to build reports which require the use of basic arithmetic operations, exponents and square roots.

Usage of software and hardware platforms

- Katherine prefers to use an electronic scientific calculator for most of her needs.
- She uses Excel as her work calculator when she needs to make reports as it allows her to perform multiple calculations simultaneously.
- She has never used the command line and does not want to learn about it.
- A number pad is a necessity for her as she can enter values very quickly.

Calculator preferences

- She values simplicity and design over functionality as she enjoys working with software that appeals to her aesthetically.
- She would like to be able to incorporate the power of excel into a calculator and to be able to see a history of all previous calculations.

Influencers that surround the persona that may influence choices

- Design is the biggest factor that influences her choice of software and applications.

3.5 Persona 5 - Maria

Private information

- Female, age of 58
- Studies completed: High School
- Works full-time as a high school tutor
- Ottawa, Ontario
- Single mother of 3 kids who loves taking photos of food she cooks or places she visits and uploads them to her instagram/facebook. Loves watching movies and spending time with her family.

Usage of calculators and functions

- Maria uses scientific calculators on a daily basis for her teachings. She is in charge of students in grades 9, 10 and 11. Therefore, the most common functions she uses are basic arithmetic operations (+, -, *, /), pi, sqrt(x), exponential functions, logarithmic functions, and trigonometric functions (sin(x), cos(x) and tan(x)). She also teaches her students to use a scientific calculator, since many of them are doing advanced functions for the first time.

- Outside of school, she rarely uses calculators.

Usage of software and hardware platforms

- Her calculator of choice for her courses is the electronic calculator. High schoolers do not have access to their phone or laptop during school time, therefore, only the electronic calculator is permitted.

- She rarely uses the default calculator on her Windows laptop. She may use it when she is doing a PowerPoint presentation and the desktop calculator is easier to show to everyone.

- In general, she is comfortable using a smartphone, but can barely open a new tab on a computer. Her main use of the computer is to watch movies or YouTube videos which is why she prefers very simple applications, otherwise she gets confused.

Calculator preferences

- Since Maria is a senior, she needs software that is simple to use and easy to use with big buttons so that it's easier for her to see.

- She suggests that keeping the history of previous results is perfect to save time if needed in the near future.

Influencers that surround the persona that may influence choices

- Her students and colleagues may influence her. The teaching plan and method determine what kind of calculator would be chosen by her.

3.6 Persona 6 - Olivier

Private information

- Male, age of 29
- Studied completed: Master of Finance
- Works full-time as a quantitative analyst in a hedge fund.
- New York City, New York
- He adapts quickly and easily to different situations as well as softwares and systems, he is very comfortable with technology.
- Loves playing soccer and has always had a keen interest in math hence his choice of studying in finance.

Usage of calculators and functions

- Olivier applies mathematical and statistical methods to financial and risk management problems. He develops and implements complex models to make financial and business decisions. He uses many advanced functions such as x^y, $arccos(x)$, ab^x, $log_b(x)$, $\Gamma(x)$, MAD, σ, SD, $sinh(x)$, $\sqrt{x}$, etc. to perfect his model. Basic arithmetic is also used to calculate prices or taxes.

Usage of software and hardware platforms

- Olivier is familiar with a variety of statistical software such as SPSS, Stata, SAS, R and Matlab. As the “rocket scientists of Wall Street” , their model needs to be precise and accurate, since it is used to help make financial decisions that involve billions of dollars. He has high computer literacy and knows how to program. He is really familiar with the command-line and builds small programs to perform data-mining and data cleaning.

- For basic arithmetic functions, he is more inclined towards phone calculators and electronic calculators.

Calculator preferences

- For Olivier, functionality is the most important as he uses complex functions on his calculators on a daily basis.

- The most important features that he is looking for: Correctly displaying fractions and long computations.

Influencers that surround the persona that may influence choices

- His classmates, teachers and colleagues may influence him. And if there is any better choice of calculator that is suggested by those influencers, he would definitely consider buying it.

3.7 Persona 7 - Sarah

Private information

- Female, age of 24
- Completed her medical degree (MD)
- Works full-time as resident in internal medicine.
- Lives in Vancouver with her boyfriend
- She is a quick learner and sociable. In her spare time, she likes to go to the zoo as she has always loved animals. Actually, she hesitated between going to Medical School or Veterinary school. Whenever she has time she volunteers in animal shelters. She likes taking care of people and animals and that is her passion. She also enjoys girl's nights out during the weekends .

Usage of calculators and functions

- A basic calculator comes in handy when Sarah gives a prescription and must ensure that the dose is appropriate for the patient's weight. A calculator with basic arithmetic operations is sufficient.
- A more complex calculator is necessary when she is doing research. She often takes part in research projects to further her knowledge about the human body. She frequently uses trigonometric functions, exponential

functions, logarithmic functions, mean absolute deviation (MAD), and standard deviation (SD) to analyze and predict stem cells development over time. Precision and accuracy are important for her work, since cells are measured in microns.

Usage of software and hardware platforms

- For basic calculations, Sarah uses mainly the calculator on her phone since it is handy and she always carries it with her. This calculator can also expand to a scientific calculator when she needs more functions, such as the exponential function. When she is on her laptop, she will also use the default calculator found in Mac OS.
- For her research, she uses R-studio (a programming language for statistical computing) to perform complex calculations and analyze trends in her datasets. It is much more efficient than a basic or scientific calculator, since it can receive thousands of inputs with only one small line of code. Sarah has no prior programming experience. She learned to use R-studio through Google and Stack Overflow.

Calculator preferences

- As she has to deal with patients and their health for her, accuracy and functionality is way more important than having a fast and " extravagant (design wise)" calculator.
- She loves the simplicity of the calculator on her phone and the possibility to expand to more functions when needed.
- She finds it extremely important to have a good help section in a mathematical software, since it will help people like her, with no prior programming experience, to use the software with more ease.

Influencers that surround the persona that may influence choices

- Her fellow coworkers in her medical residency can influence her decision when it comes to a calculator as they all daily make use of it.

Use Cases

4.1 Use Case Description

Actors:

1. Student
2. Game Developer
3. Hiring Manager
4. Internist
5. Tutor
6. Quantitative Analyst

Use Cases:

- Enter any mathematical expression.
- View the help screen.
- Reset all variable values to 0.
- View the values of all entered variables.
- Enter any constant to be assigned to a variable.
- Save last calculated result to a file.
- View calculation history.
- Enter any decimal number to convert to a fraction.

- Get last calculated result to convert to a fraction.
- Check if entered mathematical expression is valid.
- Split mathematical expression by highest precedence.
- Calculate each split mathematical expression using ETERNITY's mathematical functions.
- Save the result for use in history.
- View the calculated result.
- Close the application.

4.2 Structural Relationships

- Include: Let there be 2 use cases, every instance of the first use case must include the 2nd use case. (It is a mandatory relationship)

4.3 Classification of Actors

Primary, Concrete and Indirect Actors:

- Student
- Game Developer
- Hiring Manager
- Internist
- Tutor
- Quantitative Analyst

Secondary, Abstract and Direct Actors:

- None

Generalize Relationship:

- None

4.4 Classification of Use Cases

Informative and Concrete Use Cases:

- View calculation history.
- View the calculated Result.
- View the values of all entered variables.
- View the help screen.

Performative and Concrete Use Cases:

- Enter any mathematical expression.
- Reset all variable values to 0.
- Enter any constant to be assigned to a variable.
- Save last calculated result to a file.
- Enter any decimal number to convert to a fraction.
- Get last calculated result to convert to a fraction.
- Check if entered mathematical expression is valid.
- Split mathematical expression by highest precedence.
- Calculate each split mathematical expression using ETERNITY's mathematical functions.
- Save the result for use in history.
- Close the application.

4.5 Use Case Model

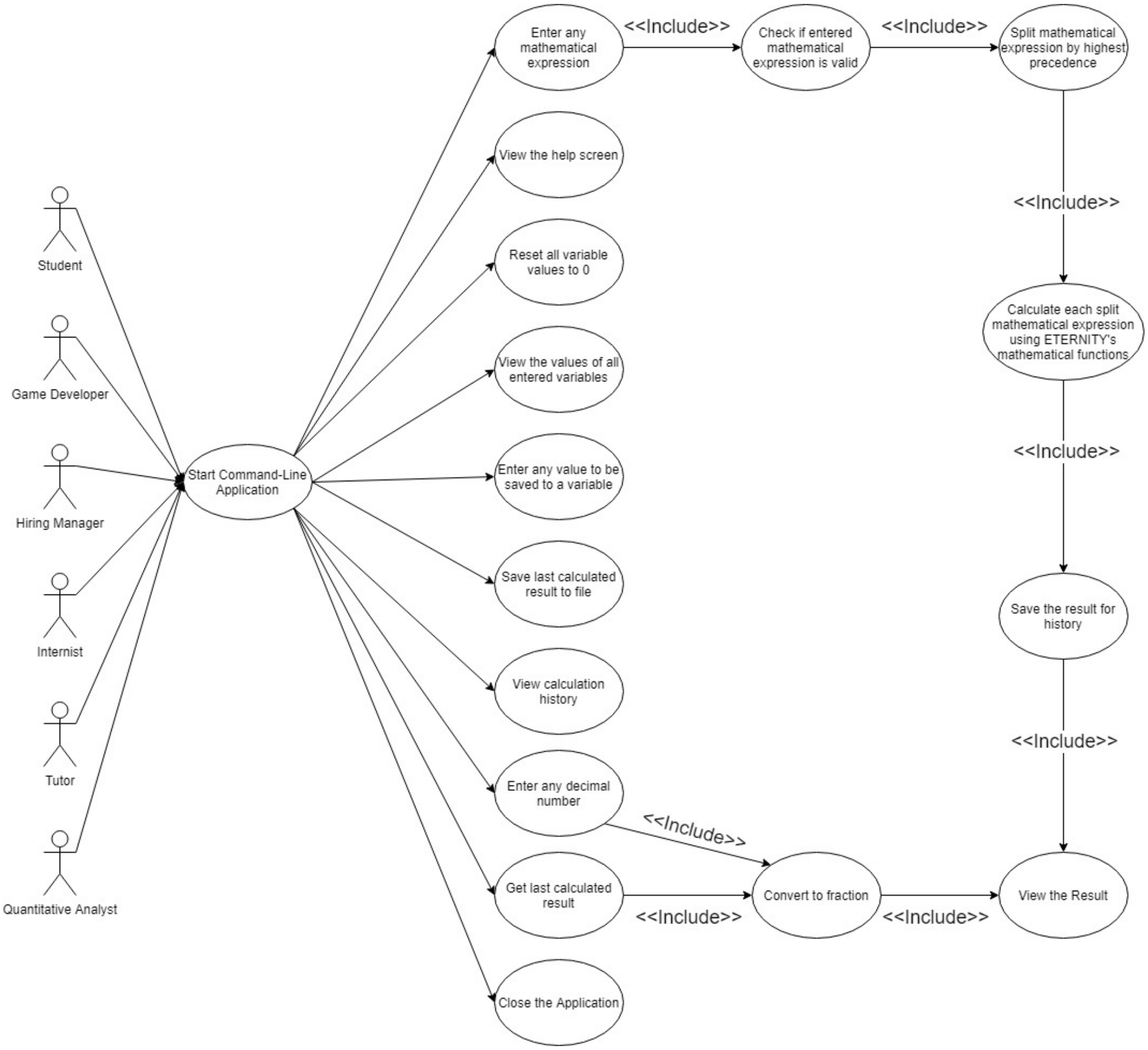

Student
Game Developer
Hiring Manager
Internist
Tutor
Quantitative Analyst
Start Command-Line Application
Enter any mathematical expression
View the help screen
Reset all variable values to 0
View the values of all entered variables
Enter any value to be saved to a variable
Save last calculated result to file
View calculation history
Enter any decimal number
Get last calculated result
Close the Application
<<Include>>
Check if entered mathematical expression is valid
<<Include>>
Split mathematical expression by highest precedence
<<Include>>
Calculate each split mathematical expression using ETERNITY's mathematical functions
<<Include>>
Save the result for history
<<Include>>
<<Include>>
Convert to fraction
<<Include>>
<<Include>>
View the Result

Figure 4.1: Use Case Model Diagram

Outline of the Strategy

5.1 Macro-architecture design

We have designed our applications in such a way that our functions can easily be integrated into our calculator with little to no work. Modularization of every component of the ETERNITY calculator was our main priority when it came to designing to the macro-architecture, we made sure that every component had a single and simple functionality such that adding or modifying components will be easy and won't break our code. Our team decided to single out the functions, the equation parser, the file writer and storage as separate components. This design follows one of the most important software designs called Separation of Concerns, where every component in a software program has a single use case.

5.2 Micro-architecture design

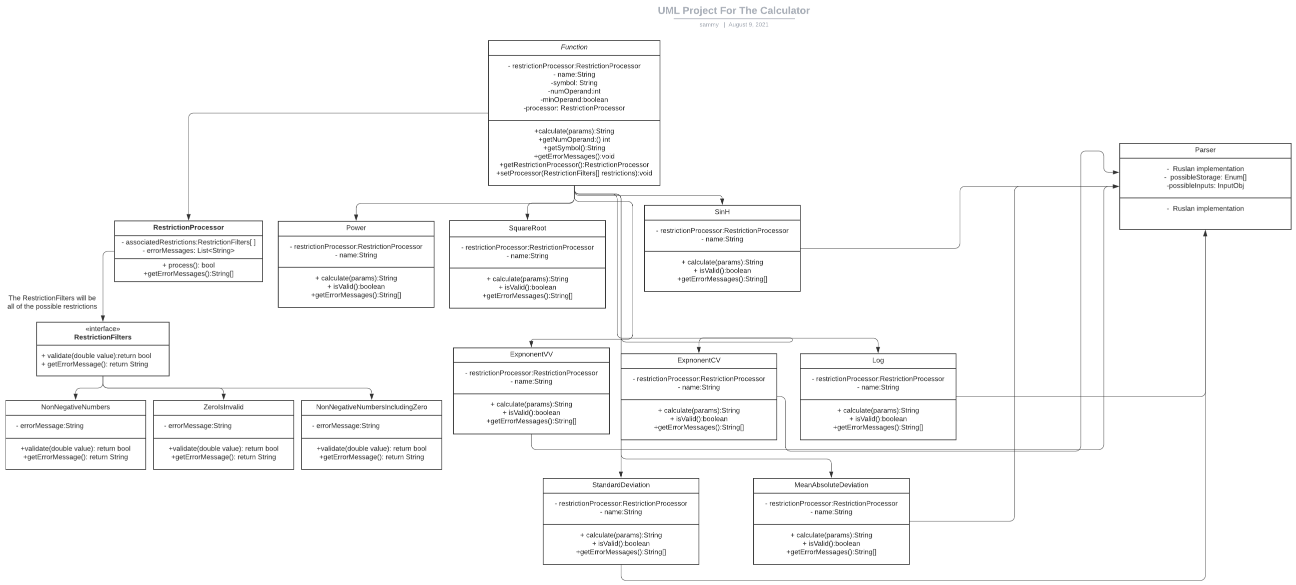
In this section we will go in-depth on how we designed the functions class.

5.2.1 Base Function Class

When creating a object oriented program, it is important to point out all possible components that can be reduced to a simple object. When it comes to functions, we can notice that all functions contain the following: a domain, a symbol, arguments and special cases. So we decided to construct a abstract class that contains all of the components that were stated previously. Having this base class allows us to have a common ground for all functions and let the concrete implementation of the function to be coded in the extended class.

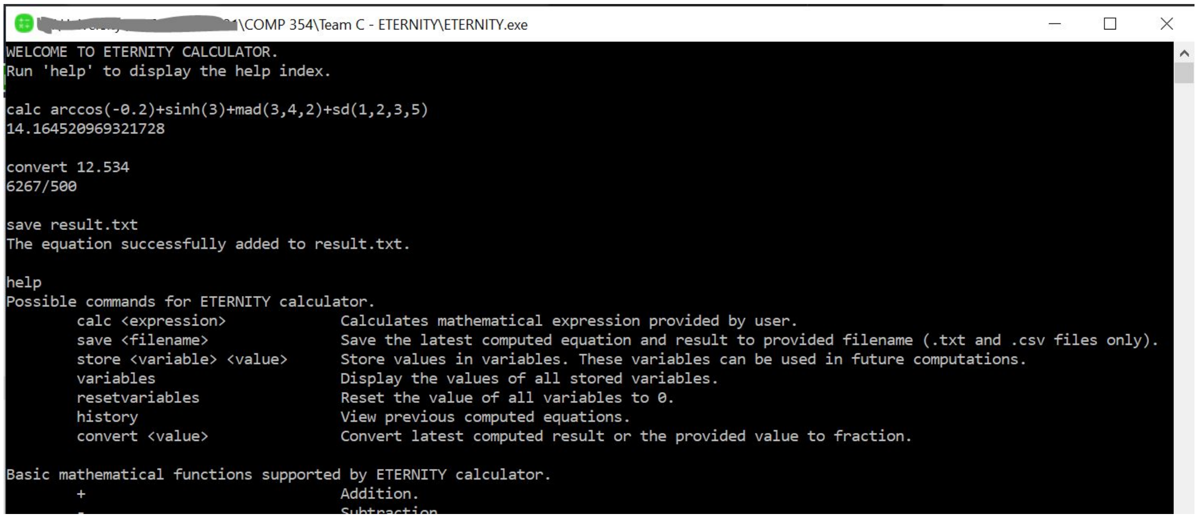
5.2.2 Filter Design Pattern

The filter design pattern is a software design pattern that allows to filter data based on a set of rules. For instance, when a user enters input to fill out their password, a password input may have a set of rules to respect and if any of the rules are not respected all issue related to the input are shown to the user. This design pattern perfectly fits the functions that we have to implement, a function is always limited to a certain domain and some explicit cases, and if those criteria are not meet we will return all of the error messages from the filters that failed to validate the data.


UML Project For The Calculator
sammy | August 9, 2021
Function
- restrictionProcessor:RestrictionProcessor
- name:String
-symbol: String
-numOperand:int
-minOperand:boolean
-processor: RestrictionProcessor
+calculate(params):String
+getNumOperand:() int
+getSymbol():String
+getErrorMessages():void
+getRestrictionProcessor():RestrictionProcessor
+setProcessor(RestrictionFilters[] restrictions):void
RestrictionProcessor
- associatedRestrictions:RestrictionFilters[ ]
- errorMessages: List<String>
+ process(): bool
+getErrorMessages():String[]
The RestrictionFilters will be all of the possible restrictions
«interface»
RestrictionFilters
+ validate(double value):return bool
+ getErrorMessage(): return String
NonNegativeNumbers
- errorMessage:String
+validate(double value): return bool
+getErrorMessage(): return String
ZeroIsInvalid
- errorMessage:String
+validate(double value): return bool
+getErrorMessage(): return String
NonNegativeNumbersIncludingZero
- errorMessage:String
+validate(double value): return bool
+getErrorMessage(): return String
Power
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
SquareRoot
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
SinH
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
Parser
- Ruslan implementation
- possibleStorage: Enum[]
-possibleInputs: InputObj
- Ruslan implementation
ExpnonentVV
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
ExpnonentCV
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
Log
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
StandardDeviation
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]
MeanAbsoluteDeviation
- restrictionProcessor:RestrictionProcessor
- name:String
+ calculate(params):String
+ isValid():boolean
+getErrorMessages():String[]

5.3 User interface design

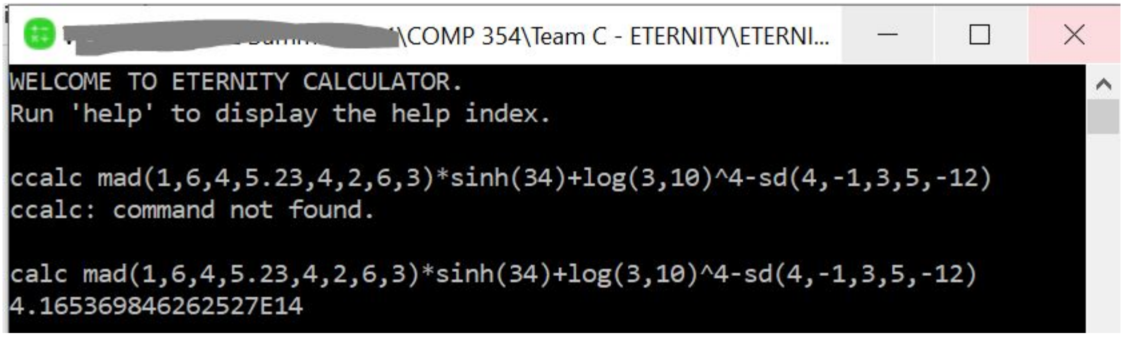
The ETERNITY calculator employs a command line interface (CLI) to interact with the user. It is a fast and efficient way to allow users to perform calculations. Users can type in the equation they wish to calculate using their keyboard or number keypad. There are nine commands that users can use to perform tasks ranging from calculating mathematical equations to saving equation results to text files. This makes the program simple, a characteristic highly requested by users in our interviews. A help section is also provided to help users get acquainted with the available commands and functions.



[redacted]\COMP 354\Team C - ETERNITY\ETERNITY.exe

```
WELCOME TO ETERNITY CALCULATOR.
Run 'help' to display the help index.

calc arccos(-0.2)+sinh(3)+mad(3,4,2)+sd(1,2,3,5)
14.164520969321728

convert 12.534
6267/500

save result.txt
The equation successfully added to result.txt.

help
Possible commands for ETERNITY calculator.
        calc <expression>            Calculates mathematical expression provided by user.
        save <filename>              Save the latest computed equation and result to provided filename (.txt and .csv files only).
        store <variable> <value>     Store values in variables. These variables can be used in future computations.
        variables                    Display the values of all stored variables.
        resetvariables               Reset the value of all variables to 0.
        history                      View previous computed equations.
        convert <value>              Convert latest computed result or the provided value to fraction.

Basic mathematical functions supported by ETERNITY calculator.
        +                            Addition.
        -                            Subtraction.
```

CLI allows users to edit their previous inputs with ease. By using the up-down arrows, users can retrieve previous entries and modify them. This way, they don't need to retype the entire expression.



```
WELCOME TO ETERNITY CALCULATOR.
Run 'help' to display the help index.

ccalc mad(1,6,4,5.23,4,2,6,3)*sinh(34)+log(3,10)^4-sd(4,-1,3,5,-12)
ccalc: command not found.

calc mad(1,6,4,5.23,4,2,6,3)*sinh(34)+log(3,10)^4-sd(4,-1,3,5,-12)
4.165369846262527E14
```

The main advantage of a CLI over a graphical user interface (GUI) is that a CLI is usually a lot faster and efficient. Thus, tasks are completed at a faster speed. A CLI also requires less memory, less CPU processing time and fewer resources. Therefore, it can be used simultaneously with other apps and software without affecting too much of its own performance and the performance of the other software.

Pseudocode of Functions

6.1 *arccos(x)* - Arccos Function

Pseudocode

```
FUNCTION Arccos
    Pass in: value
    SET arccos <- 0
    SET pi <- calculatePi()
    IF VALUE > 1 OR value < -1 THEN
        Pass Out: NotaNumber (NaN)
    ELSEIF VALUE = 1 THEN
        Pass Out: 0
    ELSEIF VALUE = -1 THEN
        Pass Out: pi
    ENDIF
    SET i <- 0
    FOR  i TO 10
        SET firstNum <- calculateFactorial(2 * i)
        SET firstDenom <- calculatePower(2, 2 * i) *
            calculatePower(calculateFactorial(i), 2)
        SET secondNum <- calculatePower(value, (2 * i) + 1)
        SET secondDenom <- (2 * i) + 1
        SET arccos <- arccos + (firstNum/firstDenom) *
            (secondNum/secondDenom)
        INCREMENT i
    ENDFOR
    SET result <- (pi/2) - arccos
    Pass Out: result
```

```
ENDFUNCTION

FUNCTION calculateFactorial
    Pass in: number
    SET factorial <- 1
    SET i <- number
    FOR  i TO 0
        SET factorial <- i * factorial
        INCREMENT i
    ENDFOR
    Pass Out: factorial
ENDFUNCTION

FUNCTION calculatePower
    Pass in: base, exponent
    SET power <- 1.0
    SET i <- 1
    FOR  i TO exponent
        SET power <- base * power
        INCREMENT i
    ENDFOR
    Pass Out: power
ENDFUNCTION
```

Technical reasons for making decisions

1. Implementing the Arccos function a loop instead of using recursion.
    - Recursion is not as efficient as loops.
    - Recursion uses more memory.
    - Recursion has a possibility of overhead (excess computation time).
2. The limit in the for loop is a constant which could be changed depending on how accurate you want the function to be.
    - A shorter limit makes results less accurate, but faster.
    - A larger limit makes results more accurate, but a lot slower.

Due to the improvements with for loops over recursion, i will be using loops. As for the limit, i chose a constant which does not present any waiting time, but still provides a pretty accurate result.

The Arccos function requires the assistance of the factorial function and the power function. They are both implemented using loops. The PI constant is required as well which will be hard coded in. The provided PI will contain a large amount of constants after the decimal to ensure the calculations used with it are still accurate.

6.2 ab^x Exponential Function

The given function contains two parts. A power calculation followed by a multiplication operation. Multiplication is a given operation and will be calculated in a single go but power is repeated multiplication and becomes a complex operation if it is a real number (not an integer). So, the main problem is to calculate power in this scenario.

1. Option 1

   - If we need to implement this function in some programming language, there are built in functions available for it and we can call them to calculate the b to the power x and then multiply it with a to get our required value.

2. Option 2

   - We can face a situation where we are working with very limited resources and there is no liberty like built-in functions for any thing like power. In such cases we will be limited to some specific operation like natural log and exponential which is also a power function but its base is always fixed to e. We can convert our function in terms of natural log and exponential using laws of logarithms. Let,

     - y = ab^x
       c = b^x →y = ac
       Taking natural log on both sides
       ln c = ln b^x
       ln c = x ln b

$$c = e^{xlnb}$$

or,

$$y = ae^{xlnb}$$

- This can be implemented using two built in functions (natural log and exponential). But there can be some scenarios where even these functions are not available like implementing in a hardware or working in some low-level language like assembly.

3. Option 3

    - Here, we will use the above-described function but implement our own functions for natural log and exponential. We will use some numerical methods to calculate the above functions. We will use arctanh sigma series expansion to approximate the function of natural log and Maclaurin series expansion for approximation of exponential function.

Pseudocode

```
FUNCTION calculate factorial
    Pass In: number n
    IF n > 2
        Pass out 1 and terminate
    Pass out n × CALL: calculate factorial of (n { 1)
ENDFUNCTION

FUNCTION get binary representation
    Pass In: number n
    INIT: i := 0, array1, array2
    WHILE n > 0
        array1(i) := n mod 2
        n := n / 2
        INCR: i := i + 1
    ENDWHILE
    DECR: i := i { 1
    INIT: j := 0
    WHILE i >= 0
        array2(j) := array1(i)
        INCR: j := j + 1
        DECR: i := i { 1
    ENDWHILE
    Pass Out: array2
ENDFUNCTION

FUNCTION power of a number to an integer
    Pass In: a and b (b must be an integer)
    INIT: ans := 1
    LET: binary := CALL: get binary representation of b
    FOR every bit in binary
        ans := ans * ans
        IF bit = 1
            ans := ans * a
        ENDIF
    ENDFOR
    Pass Out: ans
ENDFUNCTION

FUNCTION natural log
    Pass In: number x
    INIT: ans := 0, i := 0
    WHILE i < iteration_threshold*
        ans := ans + (1÷(2×i+1)) × (CALL: power of (x-1)/(x+1) to 2×i+1)
        INCR: i := i + 1
    ENDWHILE
    Pass Out: ans
ENDFUNCTION

FUNCTION exponential
    Pass In: number x
    INIT: ans := 0, i := 0
    WHILE i < iteration_threshold*
        ans := ans + (CALL: power of x to i) ÷ (CALL: factorial of i)
        INCR: i := i + 1
    ENDWHILE
    Pass Out: ans
ENDFUNCTION
```

```
FUNCTION calculate value ab^x
    Pass In: a, b, x
    INIT: t1 := CALL: natural log of b
    INIT: t2 := x × t1
    INIT: t3 := CALL: exponential of t2
    INIT: ans := t3 × a
    Pass Out: ans
ENDFUNCTION
```

* iterationthreshold is a value that will determine the accuracy of function. For high accuracy, use a high number like 100 or 150 but that will make it computationally expensive. To make the function fast, set this number to be low like 10 or 25. The function will be fast but results will not be highly accurate but will be acceptable.

Reason for Selection

We have selected option 3 because it is a complete implementation. It is not dependent on any built-in module of any language so can be implemented from scratch even in assembly language. Furthermore, we have a complete control on its speed and accuracy. We can tweak this function according to our usage scenario we can get speed from it by compromising accuracy where speed is our interest and vice versa.

6.3 $log_b(x)$ - Logarithmic Function

Two ways to implement function:

1. Using recursion
   Pseudocode - Recursion

```
FUNCTION logb
    PASS IN: two numbers x and base
    IF x > (base - 1) THEN
        PASS OUT: 1 + logb(x/base, base)
    ELSE
        PASS OUT: 0
        END FUNCTION
```

2. Using iteration
   Pseudocode - Iteration

```
        FUNCTION log
            PASS IN: two numbers x and base
            SET count:= 0
            WHILE true
                IF x > (base-1) THEN
                    x <- x/base
                    count++
                ELSE
                    PASS OUT: count
        END FUNCTION
```

3. Using ln to calculate log
   Pseudocode - Iteration

```
        FUNCTION log
            PASS IN: a number x
            IF x <= 0 THEN
                PASS OUT Double.NaN
            SET n = (x - 1) / (x + 1)
            SET result = n
            SET nQuadratic = n^2
            SET denominator = 1
            SET previousResult = 0
            WHILE result != previousResult
                previousResult = result
                denominator += 2
                n *= nQuadratic
                result += n / denominator
            PASS OUT: result
        END FUNCTION
```

```
FUNCTION log
PASS IN: a number x and a number base
    PASS OUT: return ln(x)/ln(base)
END FUNCTION
```

Technical reasons for making decisions

- Using recursion, it can make the code briefer and easy for manage, but takes more stack space and runs slower.
- Using iteration, its code is longer but it uses less stack and can run faster.
- Using iteration to calculate ln, then use the ln function to calculate log. It is way more accurate by using this method. Therefore, We used method 3 to implement the code.

6.4 *sinh*(*x*) - Hyperbolic Sine Function

Pseudocode - POW

```
PI <- 3.14159265358979323846 #PI Constant definition
E  <-   2.71828 # NUMBER e
FUNCTION POW
    PASS IN: number, exponent
    SET result <- 1
    WHILE exponent != 0
     result <- result * number
     DECREMENT exponent
    ENDWHILE
    PASS OUT: result
ENDFUNCTION
```

Technical reasons for making decisions on POW function

- I decided to go with the following algorithm as it requires less computation power then doing the same function with the use of recursion.

- For a high number of recursion calls may cause memory usage and performance issues as recursion forces to store each instance of the method called to memory which may lead to memory overflow, in java it is common to have a stack overflow for recursion that have way too many occurrences.

Pseudocode - Square Root

```
FUNCTION squareRoot
    PASS IN: number,precision
    //Will be applying the Newtons Formula to approximate its value
    val <- number
    limit <- precision + 10
    WHILE limit > precision
        result <- val - ( POW(val,2) - number) /(2*val)
        newLimit <- |result - val|
        val <-result
    ENDWHILE
    EXCEPTION
        When number < 0
            return error
    PASS OUT: val
ENDFUNCTION
```

Pseudocode - Sin

```
FUNCTION sin
    PASS IN: number
    // Bhaskara I Approximation algorithm
    result <- (16*number(PI-number))/( (call 2*POW(PI,2)) -
        (4*number*(PI-number)))
    PASS OUT: result
ENDFUNCTION
```

Technical reasons for making decisions on Sin function

- I decided to go with the Bhaskara I Approximation algorithm. This algorithm is one of the best approximation algorithms. It gives more precision and closer to the actual value of the sin function compared to the quadratic and least squares approximations algorithm.

Pseudocode - Sinh

```
FUNCTION sinh
    PASS IN: expression
    val <- 0
    IF sinh contains variable
     Val <- ‘sinh(expression)’
    ELSE
        val <- (1 - call POW(E, -2*expression))/(2 * POW(E,-expression))
    PASS OUT: val
ENDFUNCTION
```

6.5 *MAD* - Mean Absolute Deviation

Pseudocode

```
FUNCTION Mad
PASS IN: Array of real_numbers
SET mean_value <- Call: Mean(array of real numbers)
FOR every real_number IN the array
   variation <- real_number - mean_value
   positive_variation <- Call: AbsoluteValue(variation)
   sumVariation <- sumVariation + positive_variation
ENDFOR
Pass Out: sumVariation/size of array
ENDFUNCTION

FUNCTION Mean
PASS IN: Array of real_numbers
SET sum <- 0
FOR every real_number IN the array
   sum <- sum + real_number
ENDFOR
SET mean_value <- sum/size of array
PASS OUT: mean_value
ENDFUNCTION

FUNCTION AbsoluteValue
```

```
PASS IN: value
IF value < 0 THEN
    value <- value * -1
ENDIF
PASS OUT: value
ENDFUNCTION
```

Technical reasons for making decisions
Two ways to gather the inputs for the MAD function through the command-line:

1. Input separated by comma within the function: E.g.: MAD(1,2,3,4,5)
    - This method is more straightforward and allows the MAD function to be combined with another function.
    - It can be harder to use when the calculation involves a large number of inputs. However, if the inputs are already in a csv file, the user can copy paste into the command-line and thus avoids typing errors.
2. User enter the function (MAD) and program ask user for a .csv file containing all the inputs
    - Advantageous for users who needs to compute a larger number of inputs and it is already stored within a csv file.
    - The drawback is that the user must store all its inputs in a csv file before using this function.
    - Another drawback is that it would be hard to use the MAD function with another function within one mathematical expression (Would need a more complex parser).

Overall, the first method (Input separated by comma) is a preferable method of implementation.

When designing the MAD function, instead of having one single function, I separate it into 3 functions (MAD, mean and absoluteValue). By separating the computation into three functions, the program is more readable and looks more like the mathematical expression.The time complexity is O(n) and the space complexity is constant.

6.6 *SD* - Standard deviation

Pseudocode

```
FUNCTION square
    PASS IN number
    PASS OUT number * number
END FUNCTION

FUNCTION squareRoot
    PASS IN: number
    SET temp
    SET sqrt TO number / 2
    DO WHILE (temp - sqrt != 0)
        temp <- sqrt
        sqrt <- (temp + number / temp)/2
    END WHILE
    PASS OUT: sqrt
ENDFUNCTION

FUNCTION variance
    PASS IN number[]
    SET sum <- 0.0
    SET mean
    SET n <- number length
    FOR i TO n
        sum = sum + number[i]
    ENDFOR
    mean = sum / n
    SET sigma <- 0.0
    FOR i TO n
        sigma <- sigma + square(number[i] - mean)
    ENDFOR
    PASS OUT: sigma / (n)
ENDFUNCTION

FUNCTION standardDeviation
    PASS IN: number[]
    PASS OUT: squareRoot(variance(number))
ENDFUNCTION
```

Technical reasons for making decisions

- Recursion(disadvantage: recursion is normally slower, takes up more of the stack as well)
- do while loop and for loop(Better for memory and run time saving, and the code is straight forward)

Thus, we use do while loop and for loops to implement standard deviation function and other functions that are related to it.

6.7 x^y - Exponential Function

Pseudocode

```
FUNCTION exponentFunction
    PASS IN: base, exponent
    IF exponent = 0 THEN
        PASS OUT: 1
    ENDIF
    IF exponent < 0 THEN
     SET base <- 1 / base
     SET exponent <- -exponent
    ENDIF
 SET result <- base
 SET i <- 1
 FOR i TO exponent
 SET result <- result * base
    ENDFOR
    PASS OUT: result
ENDFUNCTION
```

Technical reasons for making decisions

1. Using recursion instead of an iterative approach.
    - Would have shortened the amount of code.
    - Possibility of overhead which would have been expensive in processing time and memory.
    - Could be less easy to see what's going on.
2. Using a while loop instead of a for loop
    - Would require more lines of code.
    - For loop is more compact.

Tasks

7.1 Calculations Using Multiple Functions

A user can calculate ”arccos(x) + sinh(y)” and save the result to a file, where arccos and sinh are ETERNITY functions and x and y are constants.

7.2 Converting to a Fraction

A user can enter any decimal ”x” and and have that decimal converted to a fraction (ex: x→a/b) which can then be saved to a file. A user can also get the last calculated result and convert that into a fraction which can be saved to a file.

7.3 Display Calculated Equations

A user can compute several mathematical expressions using ’calc’ command, then use the ’history’ command to view past calculations.

7.4 Using Custom Variables

A user can store any value they want to a variable (ex: 4.2→X) which can then be used in any ETERNITY mathematical function (ex: arccos(X)), as well as look at their list of assigned variables by simply typing ’variables’ command.

7.5 Viewing What ETERNITY has to Offer

A user can look at the help screen to see an entire list of functions and inputs that ETERNITY supports as well as providing examples of how they can be used in order to help a user better understand how the application functions.

7.6 Combining different Functions

A user can calculate "x(y(a^b))" and save the result to a file, where x and y are both ETERNITY's mathematical functions, and a and b are constants. (ex: arccos(sinh(0.2^2)))

7.7 Comparison using many Functions

A user can enter any number of ETERNITY's functions followed by a comparison symbol (ex: > or <) and then any number of ETERNITY's functions to get a result displaying how the left side compares against the right side. (ex: sinh(arccos(0.1)) > sinh(2))

Source Code Review Results

Our team uses Git/GitHub to manage our source code and facilitate collaboration. We have two main branches: ‘main’ and ‘staging’. The main branch reflects a production-ready state. Therefore, when the program is completed, it will be pushed to the main branch. On the other hand, the staging branch tracks all the latest delivered development changes. When a team member needs to work on a component of the program, he will branch off from the staging branch and perform a pull request into the staging branch once he is done coding. This is where the source code review becomes crucial. We need to make sure that all code that is merged into the staging branch is completely functional.
For every pull request, we assign 3 reviewers to look over the code. They can either approve the code or suggest possible changes. The following are the elements that our reviewers check for.

- **Design:** Good class design allows better code reusability, safety, abstraction, and maintenance. The class should integrate well with the overall structure of the program. Data, such as attributes, should be made private. Getters and setters are implemented, if necessary, to allow other classes to access it. Methods are all related to the abstraction represented by the class.

- **Functionality:** The code needs to work as intended. We need to ensure that the output of all methods are correct and identify discrepancies. We also need to foresee potential misuse and implement corresponding exception handling (e.g.: try/catch) and restrictions.

- **Complexity:** The code should not be overly complex to read and understand. Methods should not be too long either. Lengthy methods should be broken down and make use of helper methods. It allows a better readability of code as well. Simpler solutions with the same degree of accuracy and efficiency should also be favored.

- **Naming:** Variables, attributes, methods, and classes all need good names that reflect their use. It allows everyone to better read and understand the source code.
- **Comments:** Comments should clearly reflect the reasoning behind a piece of code. However, we need to make sure that there are not too many comments, since they can impair code readability.
- **Scrutiny:** Every line of code should be scanned over and make sure we understand what it is doing.

This process helps us work as a group and ensure that code present in the staging branch is readable, maintainable and works as intended. It also helps us decrease the number of merge conflicts.

Our source code review process is inspired by Google's Engineering Practices documentation. [9]

Test results

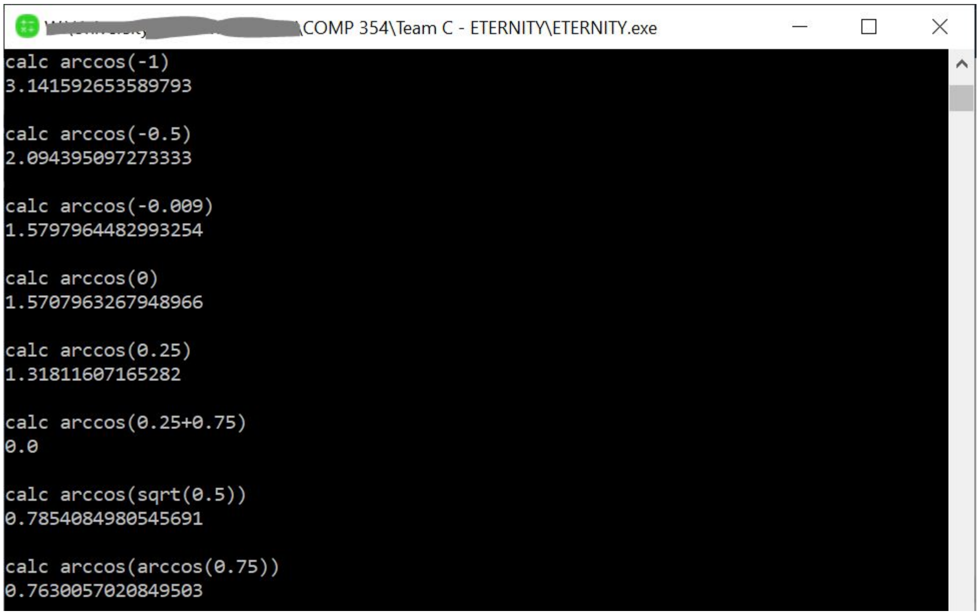
All mathematical functions have their own unit test to ensure that the output is correct.
We have also tested our program manually to ensure that the outcome (in terms of results, error handling, etc.) is what we expect. We compare our results with results from electronic calculators and online calculators such as Symbolab and Alcula [10, 11].

arccos function

Calculation result from electronic calculator.

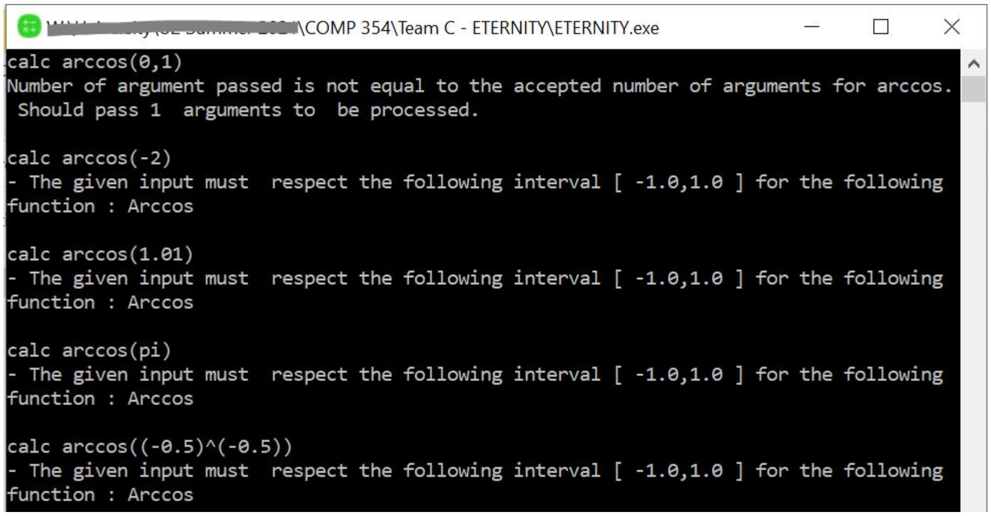
$arccos(-1) = 3.141592654$
$arccos(-0.5) = 2.094395102$
$arccos(-0.009) = 1.579796448$
$arccos(0) = 1.570796327$
$arccos(0.25) = 1.318116072$
$arccos(0.25 + 0.75) = 0$
$arccos(sqrt(0.5)) = 0.785398163$
$arccos(arccos(0.75)) = 0.763045926.$

Calculation result from ETERNITY.



```
calc arccos(-1)
3.141592653589793

calc arccos(-0.5)
2.094395097273333

calc arccos(-0.009)
1.5797964482993254

calc arccos(0)
1.5707963267948966

calc arccos(0.25)
1.31811607165282

calc arccos(0.25+0.75)
0.0

calc arccos(sqrt(0.5))
0.7854084980545691

calc arccos(arccos(0.75))
0.7630057020849503
```

The arccos function accepts one argument with values between -1 and 1.



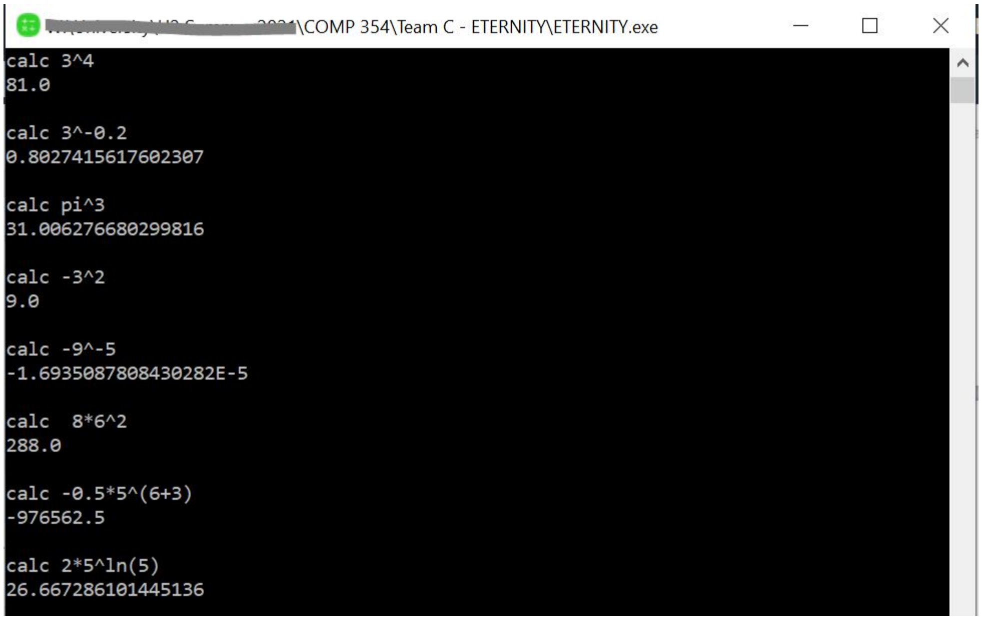
```
calc arccos(0,1)
Number of argument passed is not equal to the accepted number of arguments for arccos.
 Should pass 1  arguments to  be processed.

calc arccos(-2)
- The given input must  respect the following interval [ -1.0,1.0 ] for the following
function : Arccos

calc arccos(1.01)
- The given input must  respect the following interval [ -1.0,1.0 ] for the following
function : Arccos

calc arccos(pi)
- The given input must  respect the following interval [ -1.0,1.0 ] for the following
function : Arccos

calc arccos((-0.5)^(-0.5))
- The given input must  respect the following interval [ -1.0,1.0 ] for the following
function : Arccos
```

Exponential function: ab^x

Calculation result from electronic calculator.

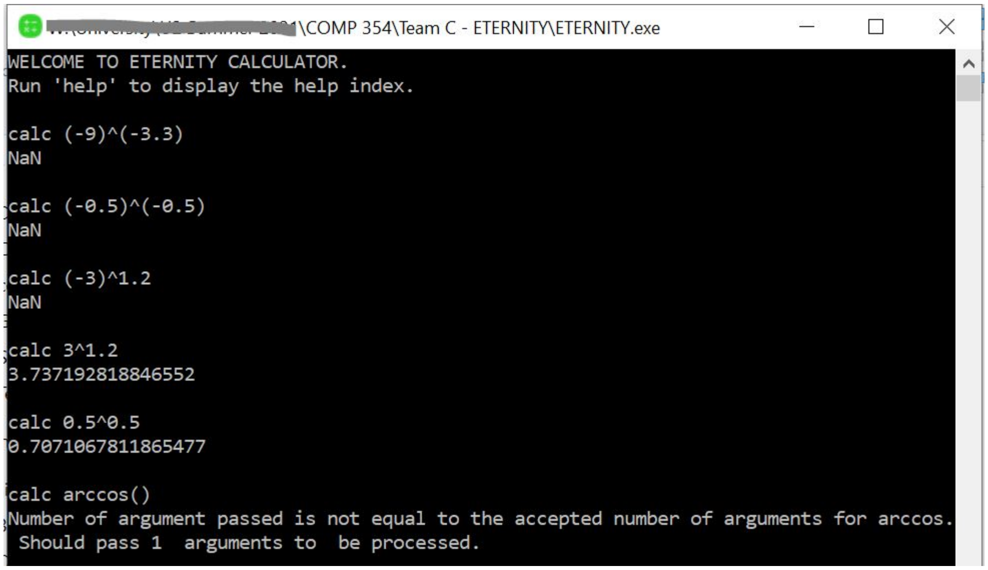
$3^4 = 81$
$3^{-0.2} = 0.802741562$
$\pi^3 = 31.00627668$
$-3^2 = 9$
$-9^{-5} = -0.000016935$
$-0.5 * 5^{(6+3)} = -976572.5$
$2 * 5^{ln(5)} = 26.6672861$

Calculation result from ETERNITY.



```
calc 3^4
81.0

calc 3^-0.2
0.8027415617602307

calc pi^3
31.006276680299816

calc -3^2
9.0

calc -9^-5
-1.6935087808430282E-5

calc  8*6^2
288.0

calc -0.5*5^(6+3)
-976562.5

calc 2*5^ln(5)
26.667286101445136
```

In many calculators, a special case of the exponential function b^x, where $b < 0$ and x is not an integer, will result in ”ERROR”. This is due to the fact that the result is often a complex number $(a + b * i)$, thus it cannot be represented in floating-point arithmetic. For these situations, we print ’NaN’.



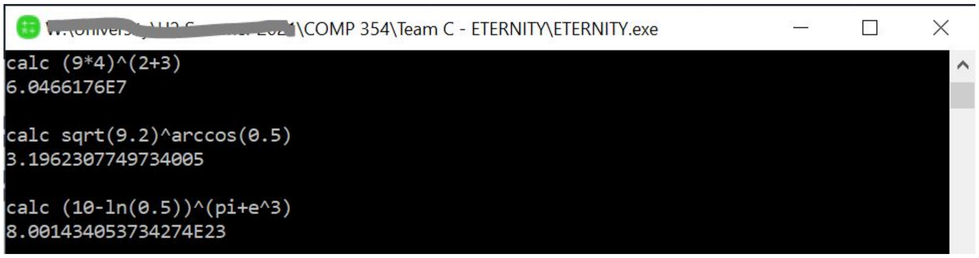
```
WELCOME TO ETERNITY CALCULATOR.
Run 'help' to display the help index.

calc (-9)^(-3.3)
NaN

calc (-0.5)^(-0.5)
NaN

calc (-3)^1.2
NaN

calc 3^1.2
3.737192818846552

calc 0.5^0.5
0.7071067811865477

calc arccos()
Number of argument passed is not equal to the accepted number of arguments for arccos.
 Should pass 1  arguments to  be processed.
```

Exponential function: x^y

Calculation result from electronic calculator.

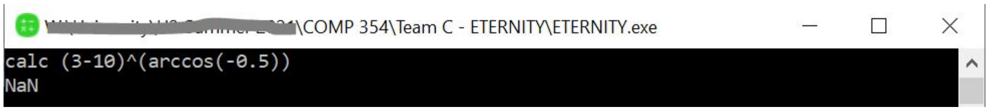
$$(9 * 4)^{(2+3)} = 60466176$$
$$sqrt(9.2)^{arccos(0.5)} = 3.19623$$
$$(10 - ln(0.5))^{(\pi+e^3)} = 8.001434053 * 10^23$$

Calculation result from ETERNITY.



```
calc (9*4)^(2+3)
6.0466176E7

calc sqrt(9.2)^arccos(0.5)
3.1962307749734005

calc (10-ln(0.5))^(pi+e^3)
8.001434053734274E23
```

In many calculators, a special case of the exponential function b^x, where $b < 0$ and x is not an integer, will result in "ERROR". This is due to the fact that the result is often a complex number $(a + b * i)$, thus it cannot be represented in floating-point arithmetic. For these situations, we print 'NaN'.



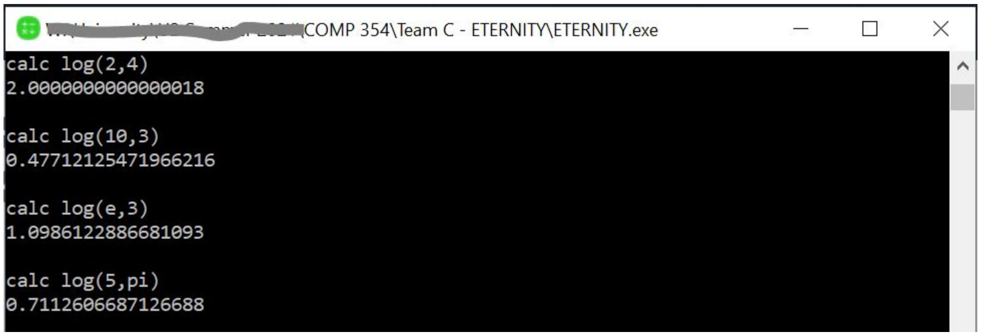
```
calc (3-10)^(arccos(-0.5))
NaN
```

Logarithmic function: $log_b(x)$

Calculation result from Symbolab (Online calculator).

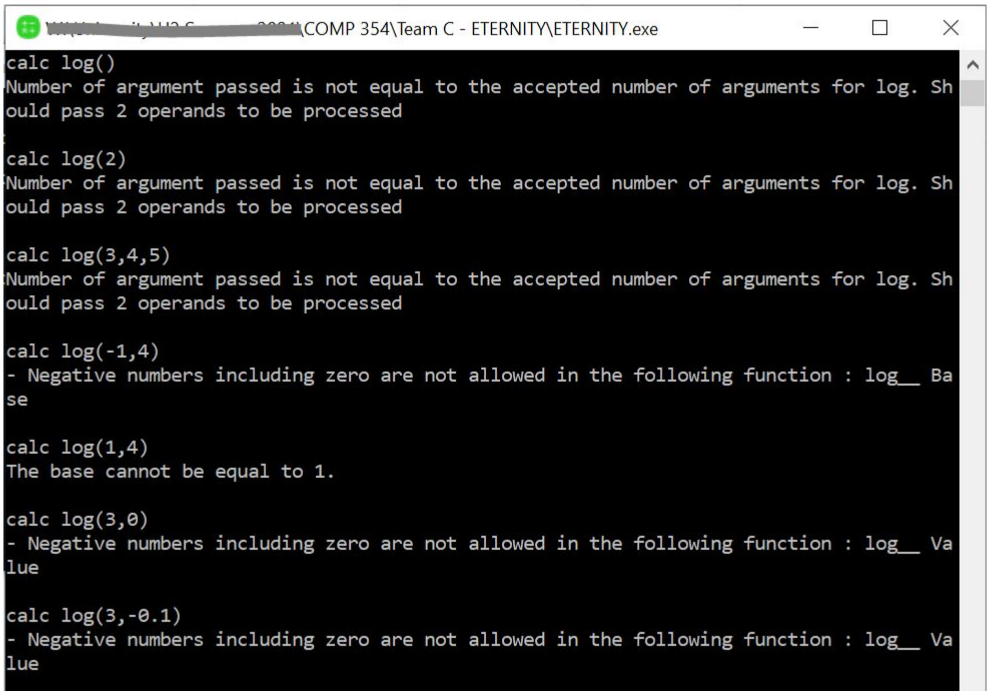
$log_2(4) = 2$
$log_{10}(3) = 0.47712..$
$log_e(3) = 1.09861..$
$log_5(pi) = 0.71126..$

Calculation result from ETERNITY.



```
calc log(2,4)
2.0000000000000018

calc log(10,3)
0.47712125471966216

calc log(e,3)
1.0986122886681093

calc log(5,pi)
0.7112606687126688
```

The logarithmic function, $log_b(x)$, accepts two arguments (log(b,x)), where $b > 0$, $b \neq 1$, and $x > 0$.



[redacted]\COMP 354\Team C - ETERNITY\ETERNITY.exe

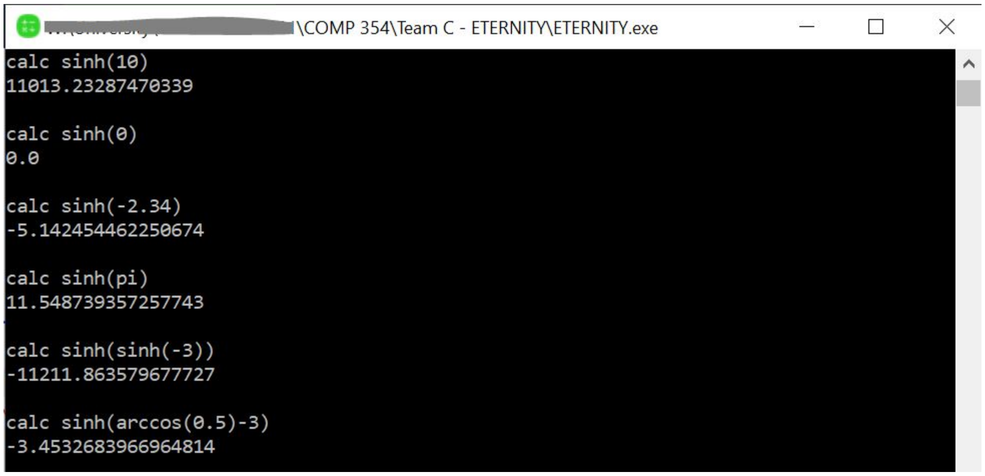
```
calc log()
Number of argument passed is not equal to the accepted number of arguments for log. Sh
ould pass 2 operands to be processed

calc log(2)
Number of argument passed is not equal to the accepted number of arguments for log. Sh
ould pass 2 operands to be processed

calc log(3,4,5)
Number of argument passed is not equal to the accepted number of arguments for log. Sh
ould pass 2 operands to be processed

calc log(-1,4)
- Negative numbers including zero are not allowed in the following function : log__ Ba
se

calc log(1,4)
The base cannot be equal to 1.

calc log(3,0)
- Negative numbers including zero are not allowed in the following function : log__ Va
lue

calc log(3,-0.1)
- Negative numbers including zero are not allowed in the following function : log__ Va
lue
```

Hyperbolic sine function: $sinh(x)$

Calculation result from Symbolab (Online calculator).

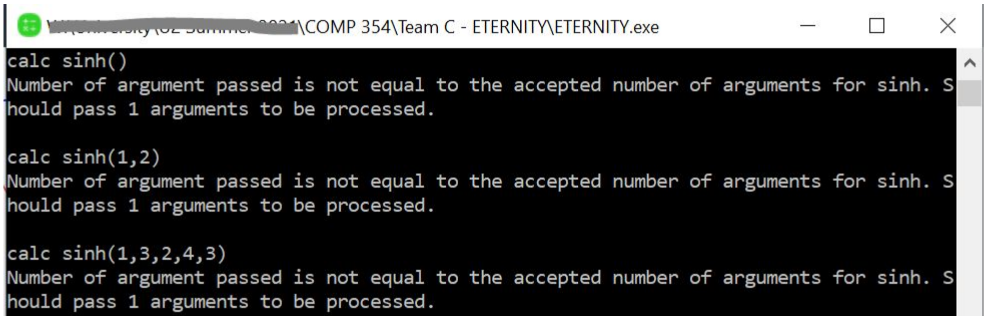
$sinh(10) = 11013.23287..$
$sinh(0) = 0$
$sinh(-2.34) = -5.14245..$
$sinh(\pi) = 11.54873.. \ldots$
$sinh(sinh(-3) = -11211.86357..$
$sinh(arccos(0.5) - 3) = -3.45326\ldots$

Calculation result from ETERNITY.



```
calc sinh(10)
11013.23287470339

calc sinh(0)
0.0

calc sinh(-2.34)
-5.142454462250674

calc sinh(pi)
11.548739357257743

calc sinh(sinh(-3))
-11211.863579677727

calc sinh(arccos(0.5)-3)
-3.4532683966964814
```

The hyperbolic sine function, $sinh(x)$, accepts one argument.



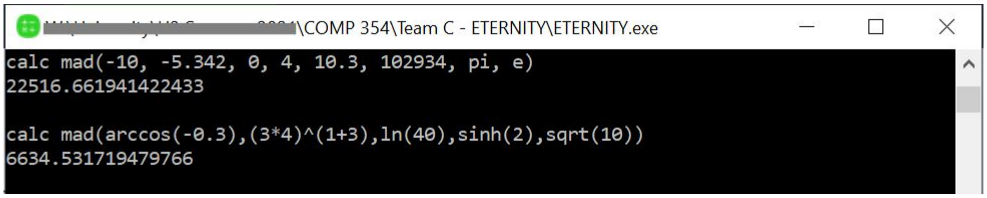
```
calc sinh()
Number of argument passed is not equal to the accepted number of arguments for sinh. S
hould pass 1 arguments to be processed.

calc sinh(1,2)
Number of argument passed is not equal to the accepted number of arguments for sinh. S
hould pass 1 arguments to be processed.

calc sinh(1,3,2,4,3)
Number of argument passed is not equal to the accepted number of arguments for sinh. S
hould pass 1 arguments to be processed.
```

Mean absolute deviation (MAD)

Calculation result from Alcula (Online calculator).

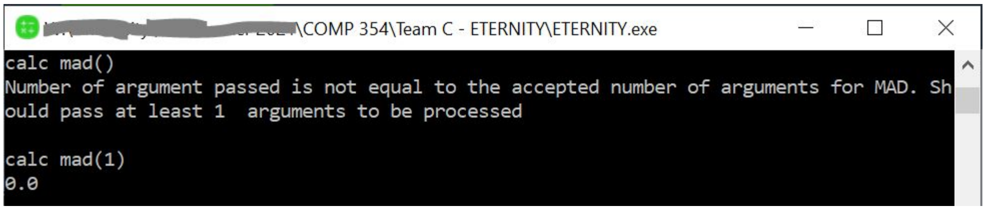
```
    MAD of (-10, -5.342, 0, 4, 10.3, 102934, pi, e) = 22516.661941422
    MAD of (arccos(-0.3),(3*4)^(1+3),ln(40), sinh(2),sqrt(10)) = 6634.5317195124
```

Calculation result from ETERNITY.



```
calc mad(-10, -5.342, 0, 4, 10.3, 102934, pi, e)
22516.661941422433

calc mad(arccos(-0.3),(3*4)^(1+3),ln(40),sinh(2),sqrt(10))
6634.531719479766
```

MAD accepts one or more argument.



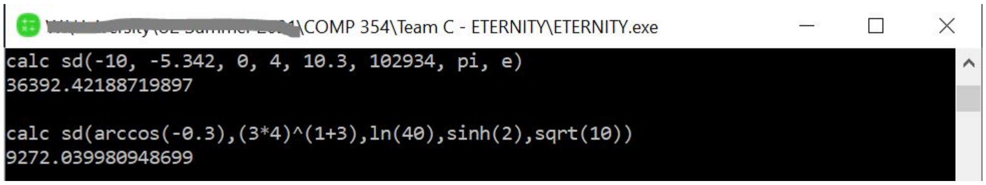
```
calc mad()
Number of argument passed is not equal to the accepted number of arguments for MAD. Sh
ould pass at least 1  arguments to be processed

calc mad(1)
0.0
```

Standard deviation (SD)

Calculation result from Symbolab (Online calculator).

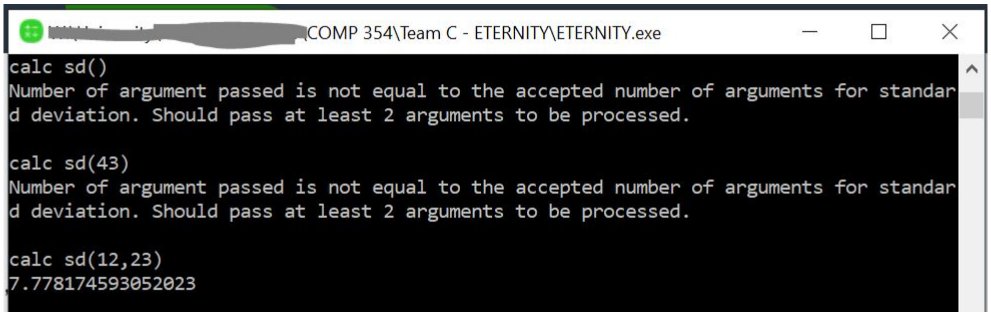
```
SD of (-10, -5.342, 0, 4, 10.3, 102934, pi, e) = 36392.42188..
SD of (arccos(-0.3),(3*4)^(1+3),ln(40), sinh(2),sqrt(10)) = 9272.03998
```

Calculation result from ETERNITY.



```
calc sd(-10, -5.342, 0, 4, 10.3, 102934, pi, e)
36392.42188719897

calc sd(arccos(-0.3),(3*4)^(1+3),ln(40),sinh(2),sqrt(10))
9272.039980948699
```

SD accepts two or more argument.



```
calc sd()
Number of argument passed is not equal to the accepted number of arguments for standar
d deviation. Should pass at least 2 arguments to be processed.

calc sd(43)
Number of argument passed is not equal to the accepted number of arguments for standar
d deviation. Should pass at least 2 arguments to be processed.

calc sd(12,23)
7.778174593052023
```

Helper Functions: Factorial (!), Natural logarithm ($ln(x)$), Square root ($\sqrt(x)$)

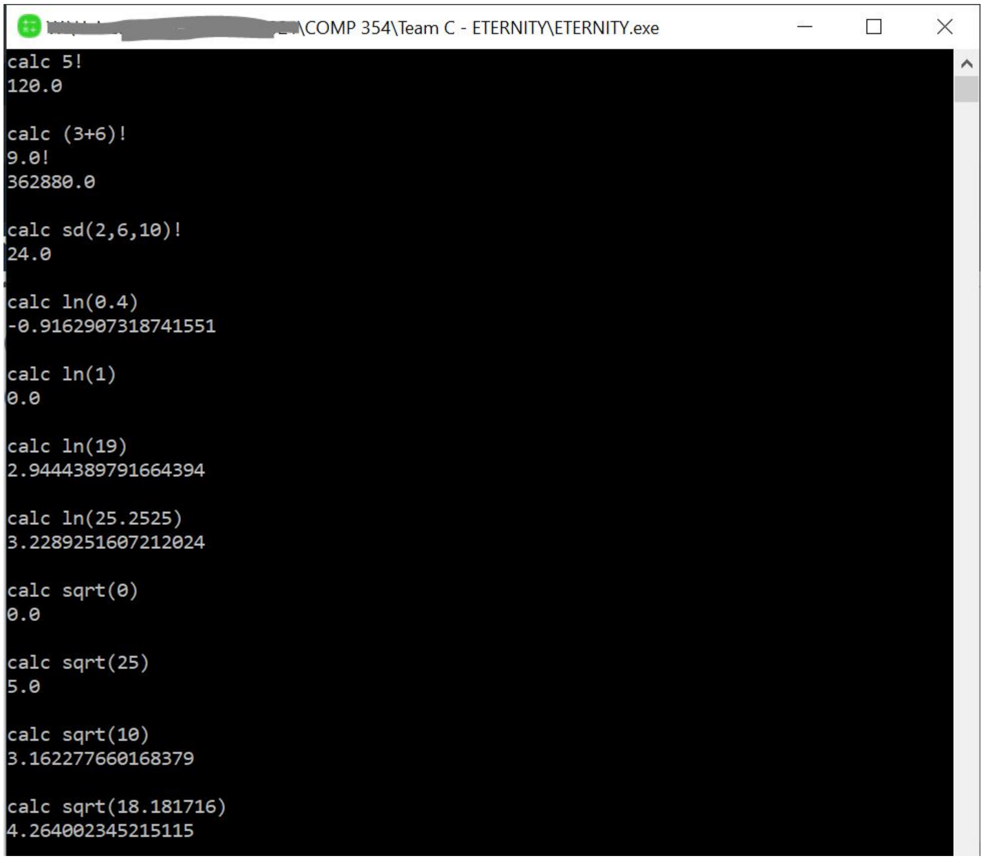
Calculation result from electronic calculator.
Need to update when factorial is fixed.

$5! = 120$
$(3 + 6)! = 263880$
$sd(2, 6, 10)! = 24$

$ln(0.4) = -0.916290732$
$ln(1) = 0$
$ln(19) = 2.944438979$
$ln(25.2525 = 3.228925161$

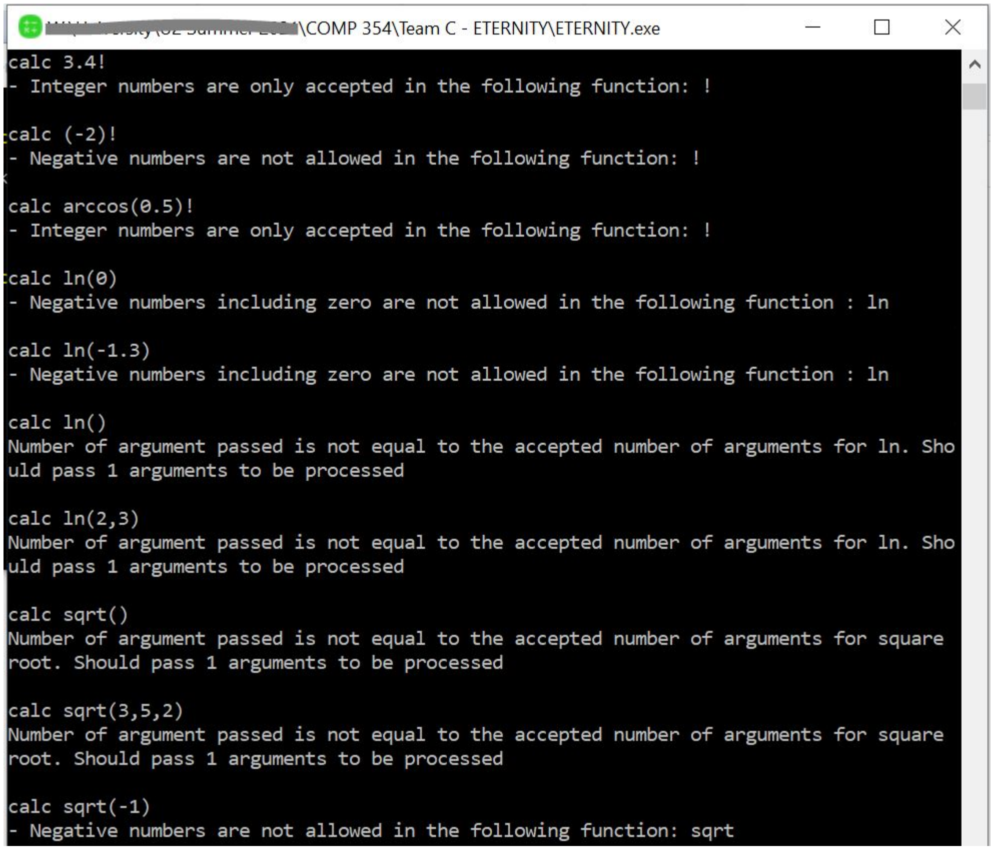
$\sqrt(0) = 0$
$\sqrt(25) =$
$\sqrt(10) = 3.16227766$
$\sqrt(18.181716) = 4.264002345$

Calculation result from ETERNITY.



```
calc 5!
120.0

calc (3+6)!
9.0!
362880.0

calc sd(2,6,10)!
24.0

calc ln(0.4)
-0.9162907318741551

calc ln(1)
0.0

calc ln(19)
2.9444389791664394

calc ln(25.2525)
3.2289251607212024

calc sqrt(0)
0.0

calc sqrt(25)
5.0

calc sqrt(10)
3.162277660168379

calc sqrt(18.181716)
4.264002345215115
```

The factorial function (!) can only be applied to positive integers.
The natural logarithmic function, $ln(x)$, accepts one argument, where $x > 0$.
The square root function, $sqrt(x)$, accepts one argument, where $x >= 0$.



[redacted]\COMP 354\Team C - ETERNITY\ETERNITY.exe

```
calc 3.4!
- Integer numbers are only accepted in the following function: !

calc (-2)!
- Negative numbers are not allowed in the following function: !

calc arccos(0.5)!
- Integer numbers are only accepted in the following function: !

calc ln(0)
- Negative numbers including zero are not allowed in the following function : ln

calc ln(-1.3)
- Negative numbers including zero are not allowed in the following function : ln

calc ln()
Number of argument passed is not equal to the accepted number of arguments for ln. Sho
uld pass 1 arguments to be processed

calc ln(2,3)
Number of argument passed is not equal to the accepted number of arguments for ln. Sho
uld pass 1 arguments to be processed

calc sqrt()
Number of argument passed is not equal to the accepted number of arguments for square
root. Should pass 1 arguments to be processed

calc sqrt(3,5,2)
Number of argument passed is not equal to the accepted number of arguments for square
root. Should pass 1 arguments to be processed

calc sqrt(-1)
- Negative numbers are not allowed in the following function: sqrt
```

Combine several functions in one mathematical expression.

Calculation result from Symbolab (Online calculator).

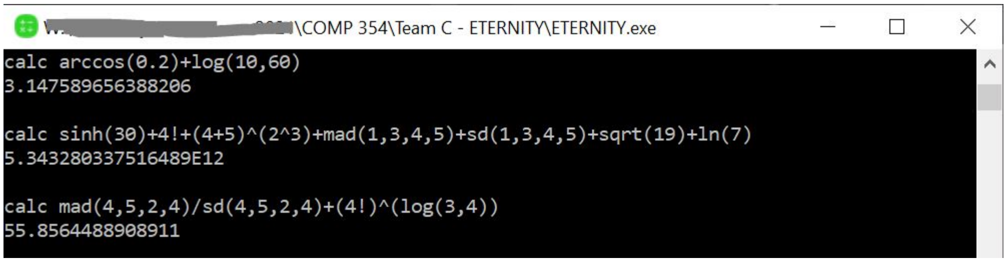
$arccos(0.2) + log_{10}60 = 3.14758..$

$sinh(30) + 4! + (4+5)^{(2^3)} + mad(1,3,4,5) + sd(1,3,4,5) + \sqrt(19) + ln(7) = 5343280337516.486$

$sinh(ln(\sqrt(arccos(0.5^0.3)))) = -0.23898..$

$mad(4,5,2,4)/sd(4,5,2,4) + (4!)^(log(3,4)) = 55.85645..$

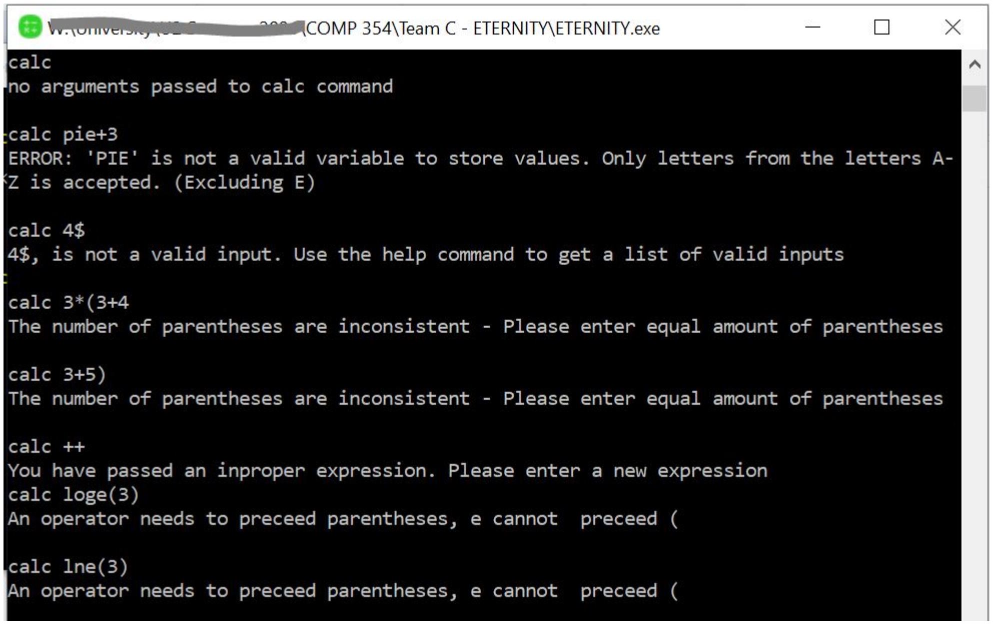
Calculation result from ETERNITY.



```
calc arccos(0.2)+log(10,60)
3.147589656388206

calc sinh(30)+4!+(4+5)^(2^3)+mad(1,3,4,5)+sd(1,3,4,5)+sqrt(19)+ln(7)
5.343280337516489E12

calc mad(4,5,2,4)/sd(4,5,2,4)+(4!)^(log(3,4))
55.8564488908911
```

Possible user errors related to 'calc' command

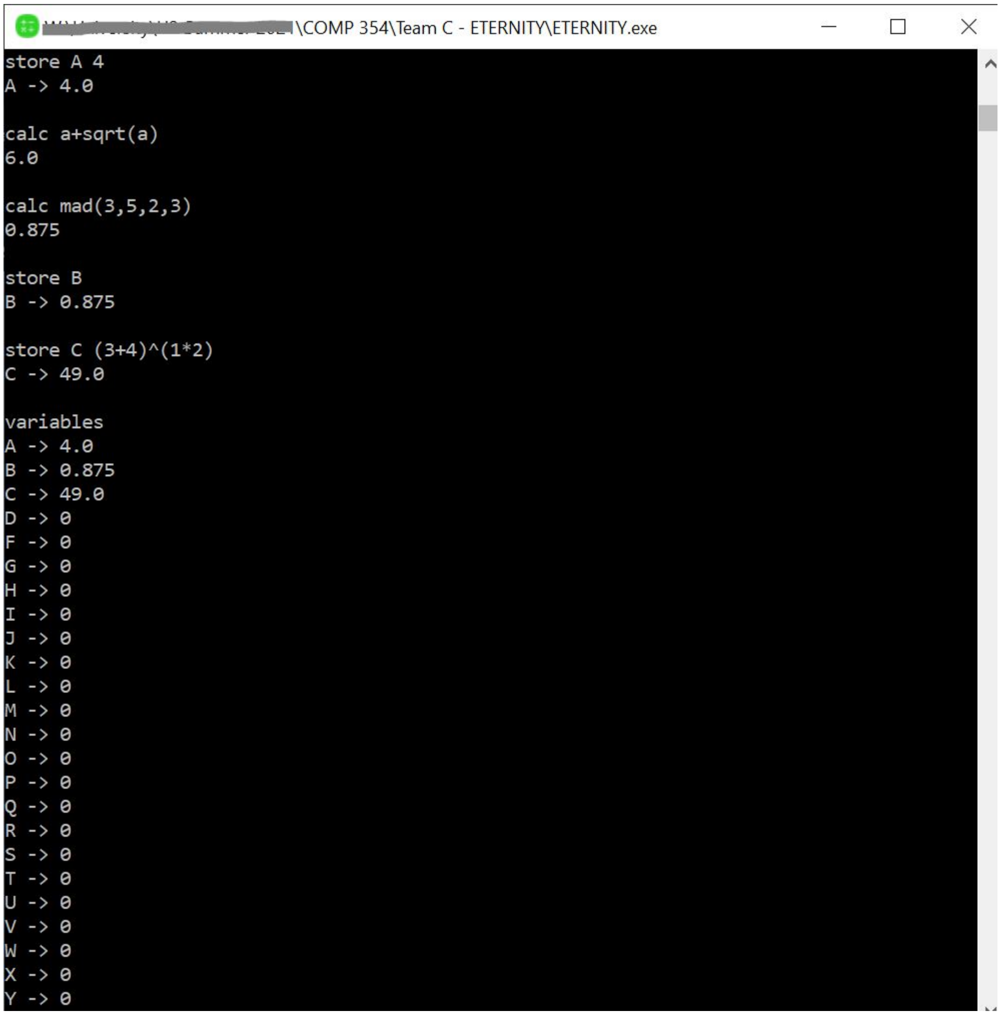
Possible user errors when computing and how it is handled, without crashing the program.



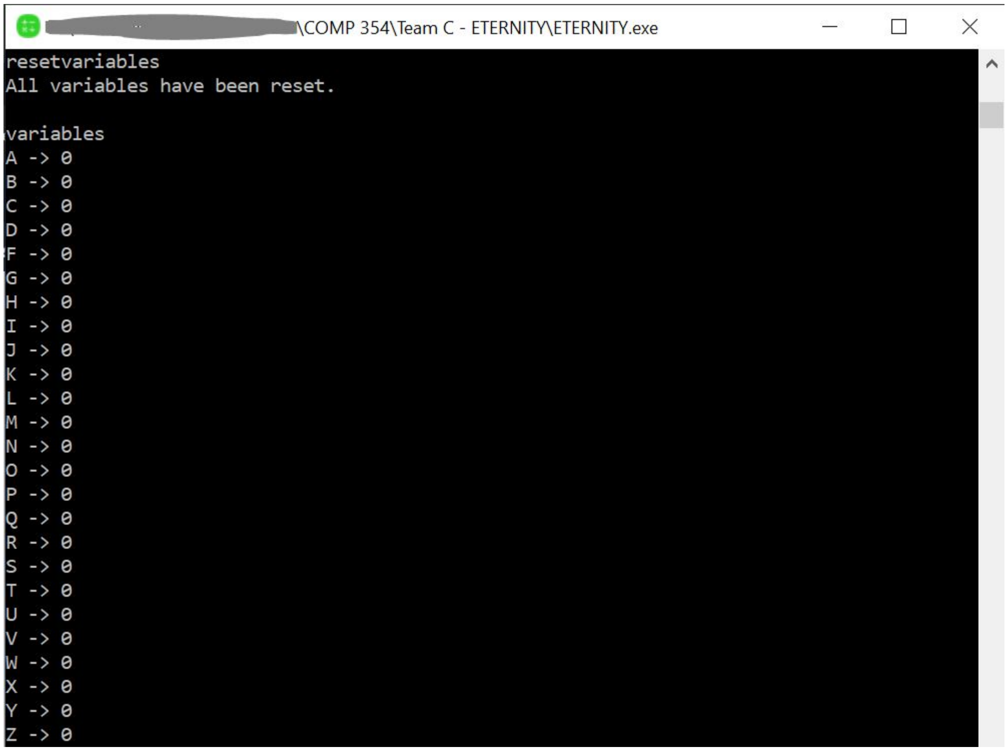
```
calc
no arguments passed to calc command

calc pie+3
ERROR: 'PIE' is not a valid variable to store values. Only letters from the letters A-
Z is accepted. (Excluding E)

calc 4$
4$, is not a valid input. Use the help command to get a list of valid inputs

calc 3*(3+4
The number of parentheses are inconsistent - Please enter equal amount of parentheses

calc 3+5)
The number of parentheses are inconsistent - Please enter equal amount of parentheses

calc ++
You have passed an inproper expression. Please enter a new expression
calc loge(3)
An operator needs to preceed parentheses, e cannot  preceed (

calc lne(3)
An operator needs to preceed parentheses, e cannot  preceed (
```

Store values in variables

Users can store values in variables and use it in another calculation later. Users can also reset the values in all variables.



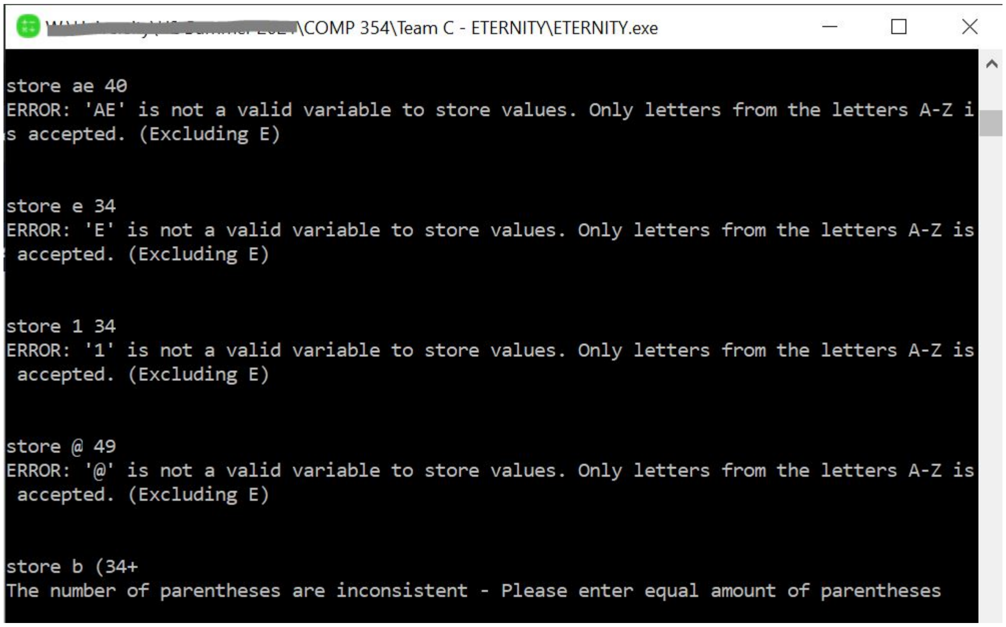
```
store A 4
A -> 4.0

calc a+sqrt(a)
6.0

calc mad(3,5,2,3)
0.875

store B
B -> 0.875

store C (3+4)^(1*2)
C -> 49.0

variables
A -> 4.0
B -> 0.875
C -> 49.0
D -> 0
F -> 0
G -> 0
H -> 0
I -> 0
J -> 0
K -> 0
L -> 0
M -> 0
N -> 0
O -> 0
P -> 0
Q -> 0
R -> 0
S -> 0
T -> 0
U -> 0
V -> 0
W -> 0
X -> 0
Y -> 0
```



[redacted]\COMP 354\Team C - ETERNITY\ETERNITY.exe

```
resetvariables
All variables have been reset.

variables
A -> 0
B -> 0
C -> 0
D -> 0
F -> 0
G -> 0
H -> 0
I -> 0
J -> 0
K -> 0
L -> 0
M -> 0
N -> 0
O -> 0
P -> 0
Q -> 0
R -> 0
S -> 0
T -> 0
U -> 0
V -> 0
W -> 0
X -> 0
Y -> 0
Z -> 0
```

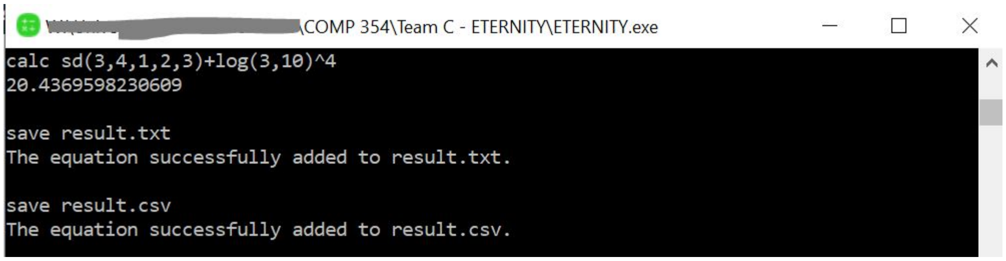
Variables can only be single alphabetical letters (A-Z, excluding E). If the value/expression is not valid, it will not be stored.



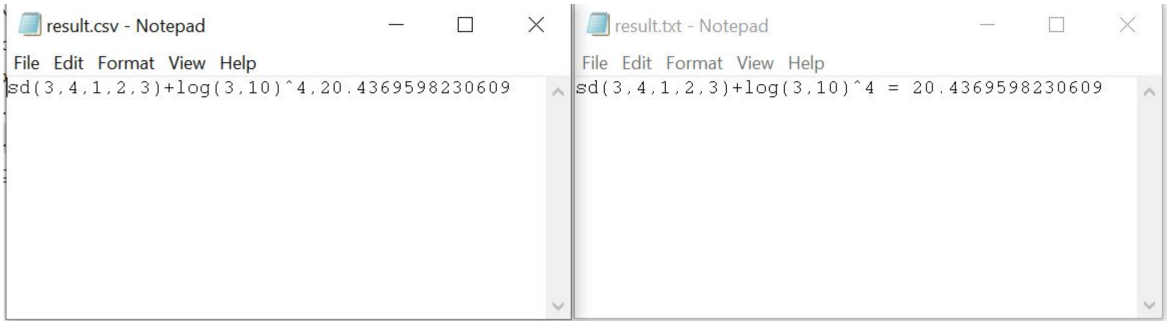
```
store ae 40
ERROR: 'AE' is not a valid variable to store values. Only letters from the letters A-Z i
s accepted. (Excluding E)


store e 34
ERROR: 'E' is not a valid variable to store values. Only letters from the letters A-Z is
 accepted. (Excluding E)


store 1 34
ERROR: '1' is not a valid variable to store values. Only letters from the letters A-Z is
 accepted. (Excluding E)


store @ 49
ERROR: '@' is not a valid variable to store values. Only letters from the letters A-Z is
 accepted. (Excluding E)


store b (34+
The number of parentheses are inconsistent - Please enter equal amount of parentheses
```

Save equations to file

Users can save mathematical expression and results to txt and csv files.



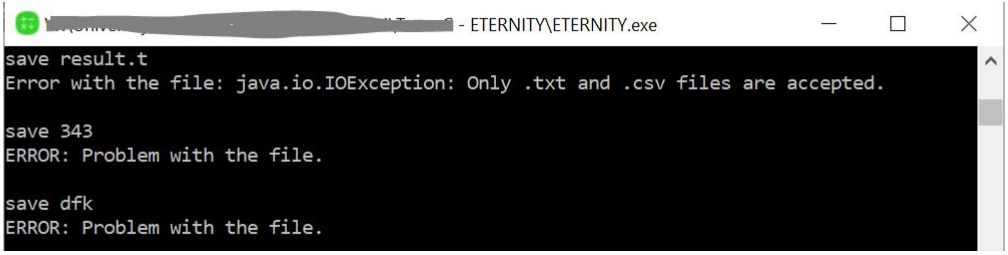
```
calc sd(3,4,1,2,3)+log(3,10)^4
20.4369598230609

save result.txt
The equation successfully added to result.txt.

save result.csv
The equation successfully added to result.csv.
```



```
sd(3,4,1,2,3)+log(3,10)^4,20.4369598230609
```

```
sd(3,4,1,2,3)+log(3,10)^4 = 20.4369598230609
```

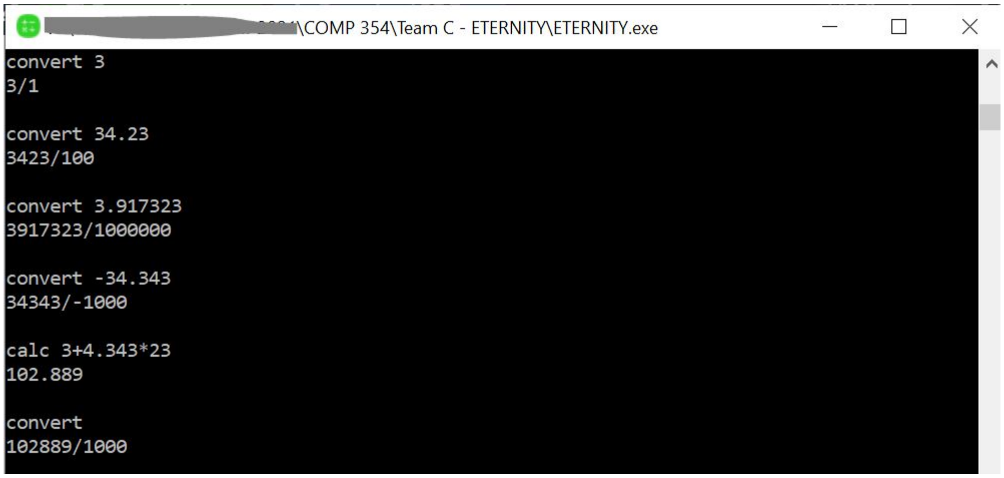
Only txt and csv files are accepted..



```
save result.t
Error with the file: java.io.IOException: Only .txt and .csv files are accepted.

save 343
ERROR: Problem with the file.

save dfk
ERROR: Problem with the file.
```

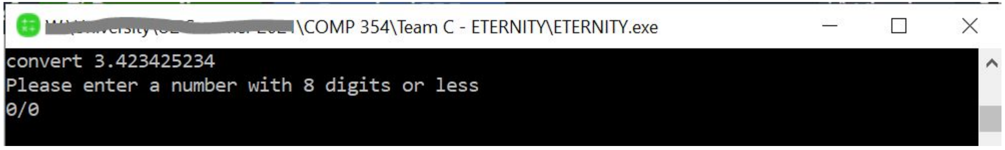
Save equations to file

Users can convert decimals to fractions.



```
convert 3
3/1

convert 34.23
3423/100

convert 3.917323
3917323/1000000

convert -34.343
34343/-1000

calc 3+4.343*23
102.889

convert
102889/1000
```

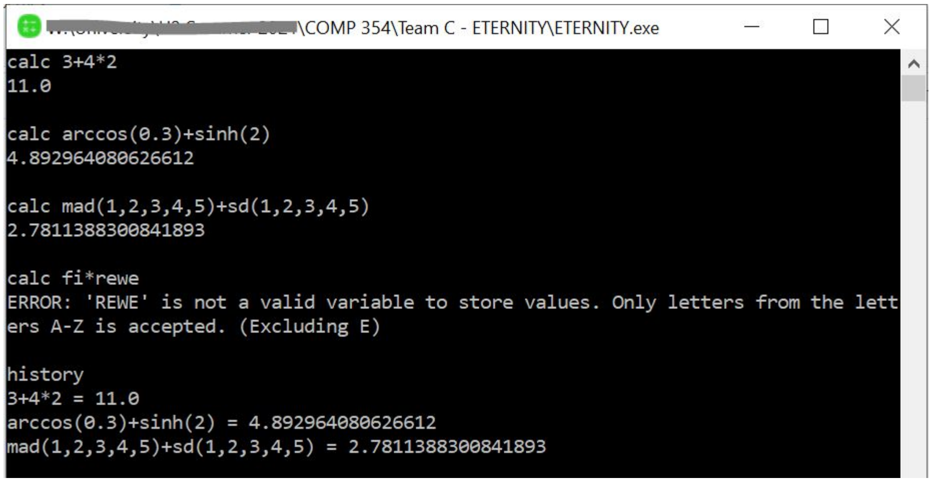
This program can only convert when decimals have 8 digis or less.



```
convert 3.423425234
Please enter a number with 8 digits or less
0/0
```

Calculation history

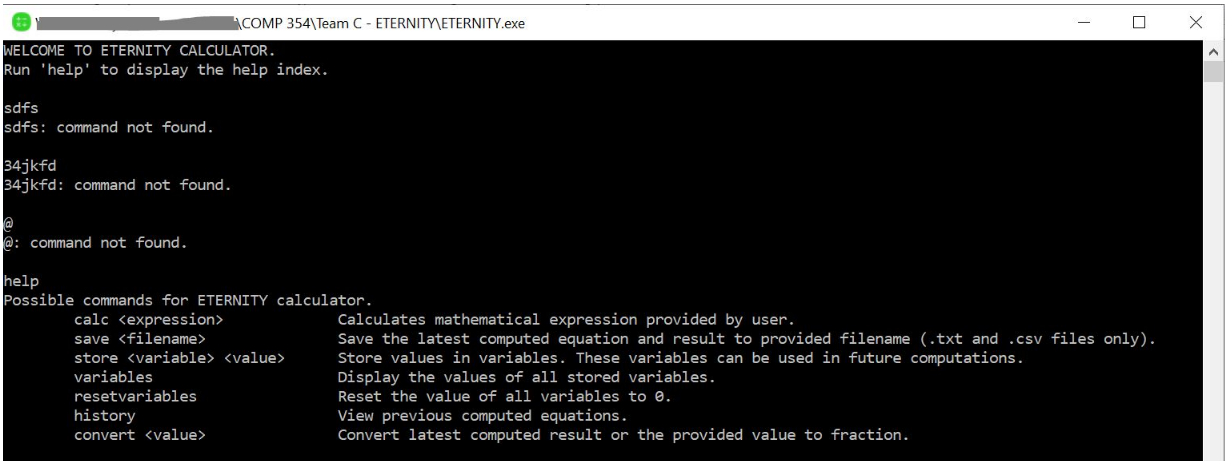
Users can view their calculation history.



```
calc 3+4*2
11.0

calc arccos(0.3)+sinh(2)
4.892964080626612

calc mad(1,2,3,4,5)+sd(1,2,3,4,5)
2.7811388300841893

calc fi*rewe
ERROR: 'REWE' is not a valid variable to store values. Only letters from the lett
ers A-Z is accepted. (Excluding E)

history
3+4*2 = 11.0
arccos(0.3)+sinh(2) = 4.892964080626612
mad(1,2,3,4,5)+sd(1,2,3,4,5) = 2.7811388300841893
```

Commands error

If the user provides a command that does not exit, an error is displayed. The user can use 'help' to see which commands are accepted by the program.



```
WELCOME TO ETERNITY CALCULATOR.
Run 'help' to display the help index.

sdfs
sdfs: command not found.

34jkfd
34jkfd: command not found.

@
@: command not found.

help
Possible commands for ETERNITY calculator.
        calc <expression>             Calculates mathematical expression provided by user.
        save <filename>               Save the latest computed equation and result to provided filename (.txt and .csv files only).
        store <variable> <value>      Store values in variables. These variables can be used in future computations.
        variables                     Display the values of all stored variables.
        resetvariables                Reset the value of all variables to 0.
        history                       View previous computed equations.
        convert <value>               Convert latest computed result or the provided value to fraction.
```

Usability Evaluation Results

In order to have our program properly tested, we had several users try out the ETERNITY calculator to see their initial reaction and how they attempt to utilize the calculator. This would provide us with valuable feedback so we could further improve our ETERNITY calculator.

10.1 What Users Liked

- **The Help Screen** - The help screen is the first option which users can see when starting the application, so a new user will most likely input "help" to see all of their available options. All testers stated that it was very detailed and helped them make get full use out of whatever they required from ETERNITY'S mathematical functions.

- **Flexibility of Input for Calculations** - Our ETERNITY calculator supports having one function within another function which can be added to another function and many other combinations. Our testers made sure to tell us how impressed they were with how many combinations of calculations they were allowed to make and still had accurate results.

- **Explanation of Errors** - When a user enters invalid data, our calculator makes sure that the user knows which mathematical function they made a mistake in, and how they could resolve the issue. It is very important for us that we do not confuse the users with cryptic messages when they make simple errors.

- **Ability to store values into variables** - Users can store values in variables and use these variables in future calculations. Users find it especially helpful when the value is a long decimal or it is the result of a very long calculation. (e.g.: A $->$ 232.14232082939, then after sinh(a) + mad(a,3,4))

10.2 What Needs Improvement

- **No Graphical User Interface** - While we made sure to make our command-line application as clean and simple as possible for users, there are people who do prefer to have a GUI as the visual representation of the calculator makes it much simpler for them to utilize.

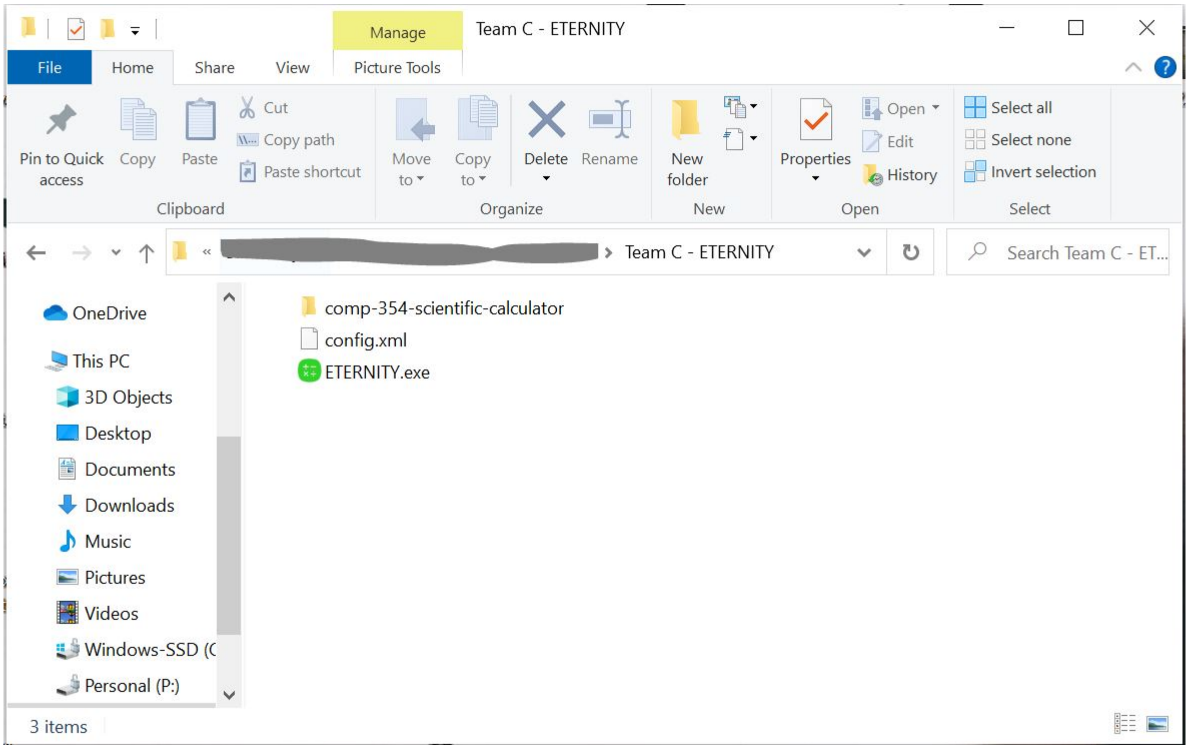
- **Memorability** - We provide many commands for our users, whether it's saving calculations to a file, converting decimals to fractions or storing variables, etc. It becomes hard for users to remember the keywords to use for each option and forces them to either remember them or constantly go back to the help screen.

10.3 Overall Evaluation Results

Overall, the base of our calculator, which is all of of mathematical functions, the error handling, etc. has been greatly appreciated by our testers. They were surprised with how flexible and accurate the calculator was. The errors they made were all caught by our advanced filters and then they were provided reasons for the errors and how they could solve them. The main problem lies with our presentation, where the testers believed that having a Graphical User Interface would make things a lot simpler and friendly to use for many other users.

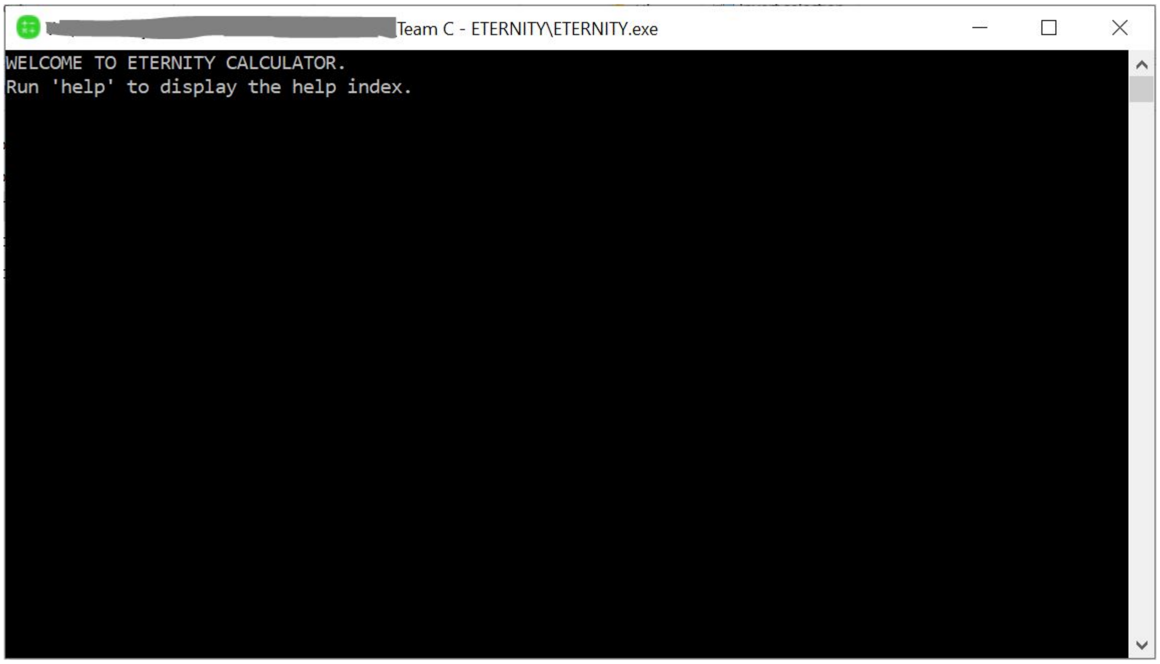
How to run ETERNITY

Open ETERNITY Calculator
Double-click the ETERNITY.exe file or ETERNITY application.



comp-354-scientific-calculator
config.xml
ETERNITY.exe

The program will launch with the following welcome message.



```
WELCOME TO ETERNITY CALCULATOR.
Run 'help' to display the help index.
```

Perform tasks with commands

ETERNITY calculator works by providing written commands through the command-line/command prompt.

Commands	Details and Examples
`help`	Display help menu and list of available commands for the program.
`exit`	Exit the program and close the command-line/command-prompt.

`calc <expression>`	Calculates mathematical expression provided by user.
	`calc 3*4`
	`calc arccos(4)+log(10,3)`
	`calc (3+3) > (5+3)`
	*At the end of this table, there is a list of supported mathematical functions and constants.

`convert <value>`	Convert latest computed result or the provided value to fraction. `convert 3.1415`
`save <filename>`	Save the latest calculated equation and result to provided filename. Text and CSV files only. `save result.txt`
`store <variable> <value>`	Store values in variables (A single alphabetical letter from A-Z, excluding E). These variables can be used in future computations. `store A 10.264`
`variables`	Display the values of all stored variables.
`resetvariables`	Reset the value of all variables to 0.
`history`	View previous computed equations.

Basic mathematical functions supported

`+`	addition
`-`	subtraction
`*`	multiplication
`/`	division
`<`	Less than. Displays 1.0 if true and 0.0 if false.

`>` Greater than. Displays 1.0 if true and 0.0 if false.

`!` Factorial.

Constants supported by ETERNITY calculator

`pi`	π: 3.14159265358979323846
`e`	e: 2.71828182845904523540

Advanced mathematical functions supported by ETERNITY calculator
'a' and 'b' are real constants. 'x' and 'y' are real variable.

`arccos(x)` Arccos function (Radian only). Accepts one argument where x is $-1 <= x <= 1$.

`ab^x` Exponential function.

`log(b,x)` Logarithmic function $log_b(x)$, where b is $b > 0$, $b \neq 1$ and $x > 0$.

`mad(x,...)` Mean Absolute Deviation (MAD). Accepts one or more arguments.

`sd(x,y,...)` Standard Deviation (σ). Accepts two or more inputs.

`sinh(x)` Hyperbolic sine function (Radian only).

`x^y` Exponential function.

Edit previous input

Since we are using a CLI, you can use the up/down arrow to retrieve previous entries and modify them.

Glossary

Abstract Class An abstract class is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed. [12]

Arc Cosine Function The arc cosine function in the form, $\arccos(x)$, is the inverse of the cosine function. It returns the angle whose cosine is given number. [2]

Arithmetic The mathematics of integers, rational numbers, real numbers, or complex numbers under addition, subtraction, multiplication, and division. [13]

Basic Calculator A device or software which can only perform basic arithmetic operations (add, subtract, multiply, divide). [14]

Basic Functions See Arithmetic.

Calculation A calculation is a deliberate process that transforms one or more inputs into one or more results.[15]

Calculator An electronic device, a mechanical device or a software used for making mathematical calculations. [16]

Command A command is a directive to a computer program to perform a specific task. It may be issued via a command-line interface, such as a shell, or as input to a network service as part of a network protocol, or as an event in a graphical user interface triggered by the user selecting an option in a menu.[17]

Command-line interface (CLI) A command-line interface processes commands to a computer program in the form of lines of text. [18]

Computer Literacy Basic nontechnical knowledge about computers and how to use them; familiarity and experience with computers, software, and computer systems. [19]

Decimal A fraction whose denominator is a power of ten and whose numerator is expressed by figures placed to the right of a decimal point.[20]

Design A design is a plan or specification for the construction of an object, a system or for the implementation of an activity or process, or the result of that plan or specification in the form of a prototype, product or process. [21]

Electronic Calculator A portable electronic device used to perform calculations, ranging from basic arithmetic to complex mathematics. [22]

Equation An equation is a statement that asserts the equality of two expression, which are connected by the equals sign "=".[23]

Error/Exception handling Error handling refers to the anticipation, detection, and resolution of programming, application, and communications errors. Specialized programs, called error handlers, are available for some applications. The best programs of this type forestall errors if possible, recover from them when they occur without terminating the application, or (if all else fails) gracefully terminate an affected application and save the error information to a log file. [24]

ETERNITY ETERNITY is a reference to the project. In this case, it will be a scientific calculator which includes the capability for parsing and evaluating mathematical expressions, where a mathematical expression is composed of one or more of the functions.

Exponential Function The exponential function of the form $f(x) = ab^x$, where b is a positive real number, and the argument x occurs as an exponent. [3]

Expression In mathematics, an expression is a finite combination of symbols that is well-formed according to rules that depend on the context. [25]

Feature A feature of something is an interesting, an important part or a characteristics of it. [26]

Field A particular branch of study or sphere of activity or interest. [27]

Filter A pattern through which data is passed. Only data that matches the pattern is allowed to pass through the filter. [28]

Filter Design Pattern A filter design pattern enables developers to filter a set of objects using different criteria and chaining them in a decoupled way through logical operations. [29]

Flexibility/Flexible code The flexibility of your code is defined by the ease with which you can modify it to fulfill some purpose you hadn't envisaged at the time you wrote it.. [30]

Function In mathematics, a function is an expression, rule, or law that defines a relationship between one variable (the independent variable) and another variable (the dependent variable). [31]

Functionality The quality of being useful, practical, and right for the purpose of which something was made. [32]

Gamma Function The gamma function of the form $\Gamma(x)$ is defined to be an extension of the factorial to complex and real number arguments. It is related to the factorial by $\Gamma(x) = (x-1)!$, for any positive integer x [5]

Graphical User Interface (GUI) A graphical user interface is an interface through which a user interacts with electronic devices such as computers and smartphones through the use of icons, menus and other visual indicators or representations (graphics). [33]

Hyperbolic Function A function related, for a real or complex variable x, to the hyperbola in a manner analogous to the relationship of the trigonometric functions to a circle. [8]

Hyperbolic Sine Function A hyperbolic function whose value of the complex number x is one-half the difference between the exponential of x and the exponential of $-x$, such that $\sinh(x) = \frac{e^x - e^{-x}}{2}$ [8]

Logarithmic Function The logarithmic function of the form $y = \log_b(x)$, that is the inverse of an exponential function (such that $b^y = x$) so that the independent variable appears in a logarithm. [4]

Mean Absolute Deviation (MAD) The mean absolute deviation of a dataset is the average distance between each data point and the mean. It gives us an idea about the variability in a dataset. [6]

Modular programming/Modularization Modular programming is a software design technique that emphasizes separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality.. [34]

Operating System (OS) Program that manages a computer's resources, especially the allocation of those resources among other programs. [35]

Scientific Calculator A device or software which can perform basic arithmetic operations (add, subtract, multiply, divide) and some advanced mathematical functions (For example, trigonometric, logarithmic, and exponential functions.) [36]

Separation of Concerns Separation of concerns is a design principle for separating a computer program into distinct sections such that each section addresses a separate concern. A concern is a set of information that affects the code of a computer program. [37]

Simple Calculator See Basic Calculator.

Software Calculator A calculator that has been implemented as a computer program, rather than as a physical hardware device. [38]

Standard Calculator See Basic Calculator.

Standard Deviation (SD) The standard deviation is a statistic that measures the dispersion of a dataset relative to its mean. The standard deviation is calculated as the square root of variance by determining each data point's deviation relative to the mean. It is commonly represented by the lower case Greek letter sigma σ. [7]

Unified Modeling Language (UML) The Unified Modeling Language is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system. [39]

Variable A variable is a container for a particular type of data (like integer, float, String and etc...). The variable name is the usual way to reference the stored value, in addition to referring to the variable itself, depending on the context. This separation of name and content allows the name to be used independently of the exact information it represents.[40]

Bibliography

[1] C. Koppe, M. van Eekelen, and S. Hoppenbrouwers. Improving student group work with collaboration patterns: A case study. In *215 IEEE/ACM 37th IEEE International Conference on Software Engineering*, pages 303–306. ACM, 2015.

[2] Math Open Reference. Operating system. https://www.mathopenref.com/arccos.html. Accessed Jun. 11, 2021. [Online].

[3] Wikipedia contributors. Exponential function. https://en.wikipedia.org/wiki/Exponential_function. Accessed Jun. 11, 2021. [Online].

[4] Merriam-Webster. Logarithmic function. https://www.merriam-webster.com/dictionary/logarithmic%20function. Accessed Jun. 11, 2021. [Online].

[5] E. W. Weisstein. Gamma function. https://mathworld.wolfram.com/GammaFunction.html. Accessed Jun. 11, 2021. [Online].

[6] Khan Academy. Mean absolute deviation. https://www.khanacademy.org/math/statistics-probability/summarizing-quantitative-data/other-measures-of-spread/a/mean-absolute-deviation-mad-review. Accessed Jun. 11, 2021. [Online].

[7] M. Hargrave. Standard deviation. https://www.investopedia.com/terms/s/standarddeviation.asp. Accessed Jun. 11, 2021. [Online].

[8] The Free Dictionary. Hyperbolic function. https://www.thefreedictionary.com/Hyperbolic+sine. Accessed Jun. 11, 2021. [Online].

[9] Google. What to look for in a code review. https://google.github.io/eng-practices/review/reviewer/looking-for.html. Accessed Aug. 08, 2021. [Online].

[10] Symbolab math solver - step by step calculator. https://www.symbolab.com/. Accessed Aug. 08, 2021. [Online].

[11] G. Arcidiacono. Statistics calculator: Mean absolute deviation (mad). http://www.alcula.com/calculators/statistics/mean-absolute-deviation/. Accessed Aug. 08, 2021. [Online].

[12] Oracle. Abstract methods and classes. https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html. Accessed Aug. 08, 2021. [Online].

[13] The Free Dictionary. Arithmetic. https://www.thefreedictionary.com/Basic+arithmetic+operations. Accessed Jun. 11, 2021. [Online].

[14] Wikipedia contributors. Calculator. https://simple.wikipedia.org/wiki/Calculator. Accessed Jun. 11, 2021. [Online].

[15] Wikipedia contributors. Calculation. https://en.wikipedia.org/wiki/Calculation. Accessed Aug. 08, 2021. [Online].

[16] Dictionary. Calculator. https://www.dictionary.com/browse/calculator. Accessed Jun. 11, 2021. [Online].

[17] Wikipedia contributors. Command (computing). https://en.wikipedia.org/wiki/Command_(computing). Accessed Aug. 08, 2021. [Online].

[18] Wikipedia contributors. Command-line interface. https://en.wikipedia.org/wiki/Command-line_interface). Accessed Jun. 11, 2021. [Online].

[19] Dictionary. Computer literacy. https://www.dictionary.com/browse/computer-literacy. Accessed Jun. 11, 2021. [Online].

[20] Lexico. Decimal. https://www.lexico.com/definition/decimal. Accessed Aug. 08, 2021. [Online].

[21] Wikipedia contributors. Design. https://en.wikipedia.org/wiki/Design. Accessed Jun. 11, 2021. [Online].

[22] Wikipedia contributors. Electronic calculator. https://en.wikipedia.org/w/index.php?title=Calculator&oldid=1023379483. Accessed Jun. 11, 2021. [Online].

[23] Wikipedia contributors. Equation. https://en.wikipedia.org/wiki/Equation. Accessed Aug. 08, 2021. [Online].

[24] TechTarget Contributor. Error handling. https://searchsoftwarequality.techtarget.com/definition/error-handling. Accessed Aug. 08, 2021. [Online].

[25] Wikipedia contributors. Expression (mathematics). https://en.wikipedia.org/wiki/Expression_(mathematics). Accessed Jun. 11, 2021. [Online].

[26] Collins. Feature. https://www.collinsdictionary.com/dictionary/english/feature. Accessed Jun. 11, 2021. [Online].

[27] Lexico. Field. https://www.lexico.com/definition/field. Accessed Jun. 11, 2021. [Online].

[28] V. Beal. Filter. https://www.webopedia.com/definitions/filter/. Accessed Aug. 08, 2021. [Online].

[29] Tutorials Point. Design patterns - filter pattern. https://www.tutorialspoint.com/design_pattern/filter_pattern.htm/. Accessed Aug. 08, 2021. [Online].

[30] Coderoom. Code for flexibility: A manifesto. https://coderoom.wordpress.com/2010/04/16/code-for-flexibility-a-manifesto/. Accessed Aug. 08, 2021. [Online].

[31] Britannica. Function. https://www.britannica.com/science/function-mathematics. Accessed Jun. 11, 2021. [Online].

[32] Merriam-Webster. Functionality. https://www.merriam-webster.com/dictionary/functionality. Accessed Jun. 11, 2021. [Online].

[33] Techopedia. Graphical user interface (gui). https://www.techopedia.com/definition/5435/graphical-user-interface-gui. Accessed Jun. 11, 2021. [Online].

[34] Wikipedia contributors. Modular programming. https://en.wikipedia.org/wiki/Modular_programming. Accessed Aug. 08, 2021. [Online].

[35] Britanica. Operating system. https://www.britannica.com/technology/operating-system. Accessed Jun. 11, 2021. [Online].

[36] Lexico. Scientific calculator. https://www.lexico.com/definition/scientific_calculator. Accessed Jun. 11, 2021. [Online].

[37] Wikipedia contributors. Separation of concerns. https://en.wikipedia.org/wiki/Separation_of_concerns. Accessed Aug. 08, 2021. [Online].

[38] Wikipedia contributors. Software calculator. https://en.wikipedia.org/wiki/Software_calculator. Accessed Jun. 11, 2021. [Online].

[39] Wikipedia contributors. Unified modeling language. https://en.wikipedia.org/wiki/Unified_Modeling_Language. Accessed Jun. 11, 2021. [Online].

[40] Wikipedia contributors. Modular programming. https://en.wikipedia.org/wiki/Variable_(computer_science). Accessed Aug. 08, 2021. [Online].