Lab Notebook

Software Tools and Technology Lab

SEBCA1191

MAULANA ABUL KALAM AZAD
UNIVERSITY OF TECHNOLOGY,
WEST BENGAL

Utech

In Pursuit Of Knowledge And Excellence

Group 24		
NAME	ROLL NO.	DEPARTMENT
Snehal Das [Leader]	30001223015	BCA
Nupur Sinha	30001223002	BCA
Nikita Debnath	30001223023	BCA
Swastik Das	30001223075	BCA
Sunit Modak	30084323005	BSc in Data Science

Index Page

I. Calculator Repository

GitHub Desktop

By Snehal Das

1 Description:

The task was divided into two parts:

1. Download GitHub desktop in your system Using this link and go through the tutorial for private repository publish.

2. Create a local repository. Build a C programme of calculator in the local repository, commit and publish it as a public repository.

2 Steps:

The task was achieved by following the below mentioned steps:-

1. Download the GitHub Desktop Application through the given link. Then basic account setup.

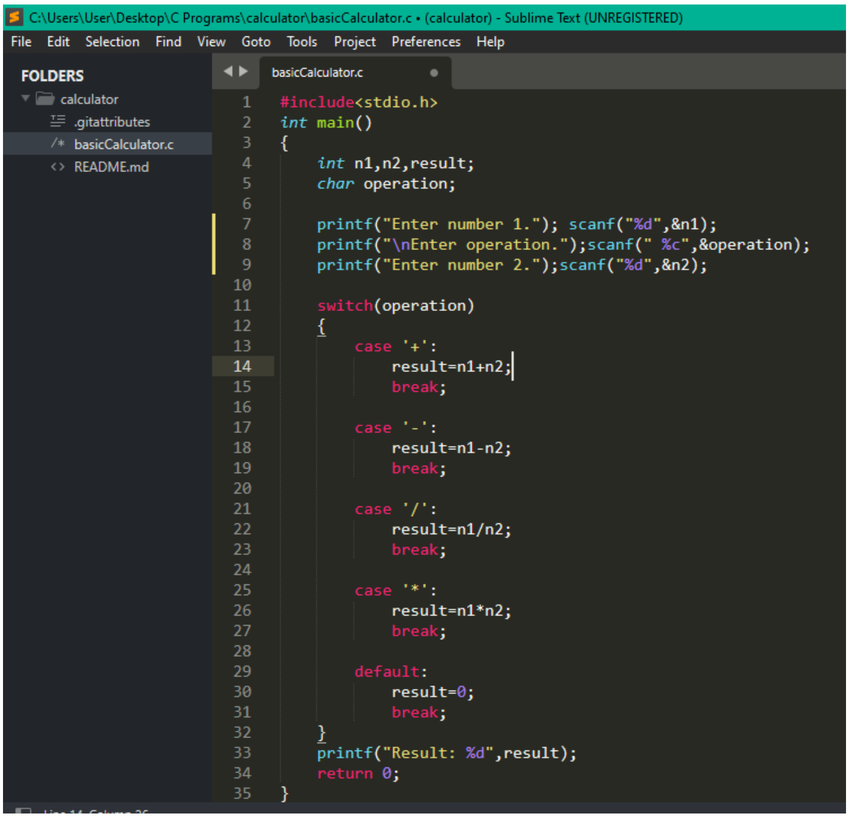
2. Using the OPEN IN SUBMILE TEXT (whichever text editor you have) option, update README file. Come back to app, and commit the change, using meaningful message.

3. In your Desktop Application, go to file $--->$ New Repository $--->$Give Name, Description, and specify local path. Check or uncheck README file and $--->$ Create Repository

4. In text editor, create a C file called basicCalculator, and write your code, save. Using similar steps, commit the change.

5. Now Publish Repository, to push the repo into remote Git server.

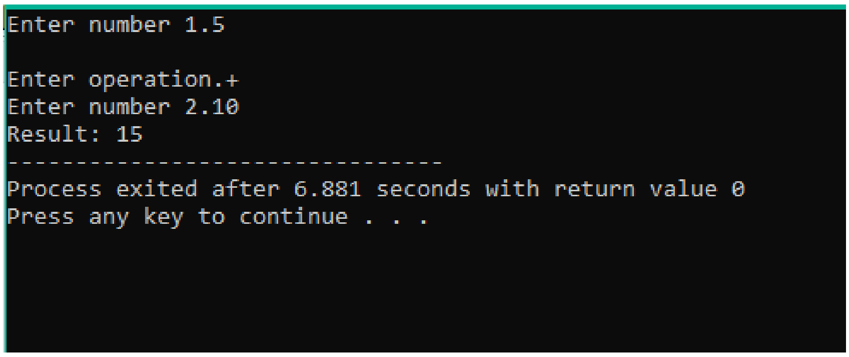
2.1 Code Snippet:


C:\Users\User\Desktop\C Programs\calculator\basicCalculator.c • (calculator) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
calculator
.gitattributes
basicCalculator.c
README.md
basicCalculator.c

```c
#include<stdio.h>
int main()
{
    int n1,n2,result;
    char operation;

    printf("Enter number 1."); scanf("%d",&n1);
    printf("\nEnter operation.");scanf(" %c",&operation);
    printf("Enter number 2.");scanf("%d",&n2);

    switch(operation)
    {
        case '+':
            result=n1+n2;
            break;

        case '-':
            result=n1-n2;
            break;

        case '/':
            result=n1/n2;
            break;

        case '*':
            result=n1*n2;
            break;

        default:
            result=0;
            break;
    }
    printf("Result: %d",result);
    return 0;
}
```

Figure 1: Source Code

This is a simple Calculator program in C, which takes in 3 inputs from the user- the two operands, and the operator. The program makes use of the SWITCH CASE statement to implement conditions, calculating and displaying the result accordingly.



```
Enter number 1.5

Enter operation.+
Enter number 2.10
Result: 15
--------------------------------
Process exited after 6.881 seconds with return value 0
Press any key to continue . . .
```

Figure 2: Output

2. How to create Matrix in LaTeX

Lab Assignment by Nupur Sinha

To create a matrix in LaTeX, we use the `amsmath` package, which provides various environments for displaying matrices. Here's a basic guide on how to create a matrix and the different options available:

- **Include the `amsmath` Package**

First, ensure you have the `amsmath` package included in your LaTeX document preamble. Add the following line:

```
\usepackage{amsmath}
```

- **Matrix Environments**

There are several environments for creating matrices, depending on how you want them formatted. Here are the most common ones:

- `matrix`: A basic matrix without brackets.
- `bmatrix`: A matrix with square brackets.
- `pmatrix`: A matrix with parentheses.
- `vmatrix`: A matrix with vertical bars.
- `Vmatrix`: A matrix with double vertical bars.

Syntax

The syntax for creating a matrix is similar across different environments. Use the `\begin{environment}` and `\end{environment}` commands to enclose the matrix content. Separate the elements in each row with `&` and
end each row with

.

Here's an example of a 2x2 matrix in each environment:

```
[
$ \begin{matrix}
    1 & 0 \\
    0 & 1
\end{matrix}
$ ]

\begin{bmatrix}   [
 $   \begin{matrix}
    1 & 0\\
    0 & 1
    \end{matrix}
 $   ]

a_{11} & a_{12} \\
a_{21} & a_{22}
\end{bmatrix}

\begin{pmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{pmatrix}

\begin{vmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{vmatrix}

\begin{Vmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{Vmatrix}
```

Explanation

- `\begin{matrix} ... \end{matrix}`: Creates a matrix with no surrounding brackets.
- `\begin{bmatrix} ... \end{bmatrix}`: Surrounds the matrix with square brackets.
- `\begin{pmatrix} ... \end{pmatrix}`: Surrounds the matrix with parentheses.
- `\begin{vmatrix} ... \end{vmatrix}`: Surrounds the matrix with single vertical bars, often used to denote determinants.
- `\begin{Vmatrix} ... \end{Vmatrix}`: Surrounds the matrix with double vertical bars, also used for determinants or norms.

III. Creating a CV Using LaTeX

LATEX File

Swastik Das

Introduction

This guide will walk you through the steps to create a CV in LaTeX and package all the source files into a zip file for submission.

2.2 Step 1: Install LaTeX Environment

If you haven't already, you need to install a LaTeX editor. Here are some options:

- **Overleaf**: An online LaTeX editor, no installation required.
- **TeXworks**: For Windows.
- **TeXShop**: For macOS.
- **Kile**: For Linux.

2.3 Step 2: Write Your LaTeX Document

1. **Open your LaTeX editor**: You can use Overleaf or an installed editor.
2. **Create a new project or file**:
   - In Overleaf, click *New Project* and choose *Blank Project*.
   - In local editors, create a `.tex` file using your LaTeX editor.
3. **Write your CV**: Write the LaTeX code for your CV, structuring it with sections like Personal Information, Education, Work Experience, etc.

2.4 Step 3: Compile the LaTeX Document

1. **Compile the `.tex` file**:
   - In Overleaf, click *Recompile*.
   - In other editors, look for the *Build* or *Compile* button.
2. **Check the output** to ensure everything looks correct.

2.5 Step 4: Gather the Necessary Files

Once the LaTeX file compiles successfully, collect these files:

- The `.tex` file (LaTeX source code).
- The compiled PDF.
- Any additional files (e.g., `.cls` files, images, etc.).

2.6 Step 5: Name the Folder

Create a folder using the following format:

`Rollno_DeptName_Firstname_CV`

For example, if your roll number is `12345`, department is `CSE`, and your first name is `John`, your folder should be named:

`12345_CSE_John_CV`

2.7 Step 6: Add Files to the Folder

Move all the required files (the `.tex` file, the PDF, and any additional files) into the folder you just created.

2.8 Step 7: Zip the Folder

- **Windows**: Right-click the folder and choose *Send to ¿ Compressed (zipped) folder*.
- **macOS**: Right-click the folder and choose *Compress*.
- **Linux**: Right-click the folder, select *Compress*, and choose the `.zip` format.

2.9 Step 8: Verify the Zip File

Before submitting, ensure that:

- The zip file name follows the format: `Rollno_DeptName_Firstname_CV.zip`.
- All necessary files (source code, PDF, and other dependencies) are included.

2.10 Step 9: Upload the Zip File

Upload the zip file to the required platform (university portal, email, etc.) for submission.

IV. My LATEX Document and Mathematical Notation

LATEX File

Nikita Debnath

2.11 Description:

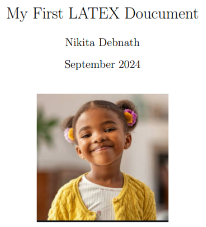
1. At first we have to create a overleaf account.
2. Then create a new blank file and the write down the code.

2.12 Steps:

1. **Basic LaTeX Document Structure:**
   Every LaTeX document starts with `{\begin{document}` and ends with `{\end{document}`. This structure defines how the document will be formatted and organized.
2. **Document Type, Title, Author, and Photo Integration:**
   To define the type of the document, we have to use `\documentclass{article}`. Use the `\title`, `\author`, and `\date` commands to display document metadata. Insert photos using the graphicx package and `\includegraphics{image_file_name.jpg}` command.
   (a) Input:

```
\documentclass{article}
\usepackage{graphicx}
\title{My First LATEX Doucument}
\date{\today}
\author{Nikita Debnath}
\date{September 2024}
\begin{document}
\maketitle
\begin{center}
\includegraphics[width=0.3\textwidth]{n.jpg}
\end{center}
\end{document}
```

   (b) Output :



My First LATEX Doucument

Nikita Debnath

September 2024

3. **Formatting and Bulleted Lists:**
   LaTeX offers various environments like itemize and enumerate for lists, and basic text can be formatted into sections and subsections with proper headers.

(a) **Section :**
The `\section{}` command creates a new section. Sections are the highest-level divisions within a document (after chapters if you're using a book class). It is usually numbered and serves as a top-level heading.

i. **Syntax:**

```
\section{Introcuction}
This is the start of a new section.
```

(b) **Subsection:**
A `\subsection{}`is a sub-division of a section. It creates a smaller heading under the main section.

i. **Syntax:**

```
\subsection{sub section}
This is a subsection within the "Introduction"section.
```

(c) **Sub-subsection:**
A `\subsubsection{}` is a further subdivision of a subsection. It is used when you need more detailed hierarchical organization.

i. **Syntax:**

```
\subsubsection{Background}
 This is a subsubsection under the "Subsection" subsection.
```

(d) **Bullet List:**
LaTeX uses the itemize environment to create unordered (bullet) lists.

i. **Syntax:**

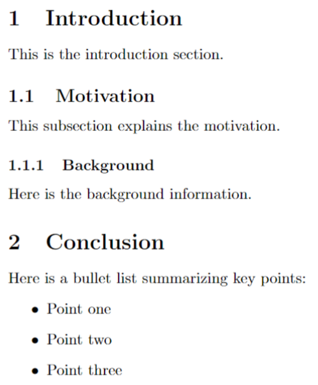
```
\begin{itemize}
\item First bullet point
\item Second bullet point
\item Third bullet point
\end{itemize}
```

This will produce a bullet list with three items.

(e) **Syntax:**

```
documentclass{article}
\begin{document}
\section{Introduction}
This is the introduction section.
\subsection{Motivation}
This subsection explains the motivation.
\subsubsection{Background}
Here is the background information.
\section{Conclusion}
Here is a bullet list summarizing key points:
\begin{itemize}
    \item Point one
    \item Point two
    \item Point three
\end{itemize}
\end{document}
```

(f) **Output:**



1 Introduction

This is the introduction section.

1.1 Motivation

This subsection explains the motivation.

1.1.1 Background

Here is the background information.

2 Conclusion

Here is a bullet list summarizing key points:

- Point one
- Point two
- Point three

4. Mathematical Expressions, Superscripts, Subscripts, and Greek Letters: LaTeX excels at typesetting mathematical symbols using `\usepackage{amsmath}` or inline display math mode with commands like `$a^b$` for superscripts and `$x_i$` for subscripts.

   (a) **Subscript:**
   Subscripts are used to write smaller text or numbers below the main text.Its writes by underscore.

   i. **Syntax:**

   ```
   $x_i$
   ```

   ii. **Output:**
   $x_i$

   (b) **Superscripts:**
   Superscripts are used to write text or numbers above the main text (for powers or exponents).

   i. **Syntax:**

   ```
   $x^i$
   ```

   ii. **Output:**
   $x^i$

   (c) **Equation Environment:**

   i. The equation environment is used for displaying equations that are centered and automatically numbered.

   A. **Syntax:**

   ```
   \begin{equation}
       x = 1
   \end{equation}
   ```

   B. **Output:**

   $$x = 1 \tag{1}$$

   ii. This will display the equation and number it.If you want to avoid numbering the equation, use the syntax -

   A. **Syntax:**

   ```
   \begin{equation*}
       x = 1
   \end{equation*}
   ```

B. **Output:**

$$x = 1$$

(d) **Align Environment, Fraction, Square-root:**

i. The align environment is used for aligning multiple equations. It allows you to align equations at the & symbol. To create fractions, use the `\frac{numerator}{denominator}` command.You can create square roots using the `\sqrt{}` command. If you want an nth root, use `\sqrt[n]{}`.

A. **Syntax:**

```
\begin{align}
x + y &= 5 \\
g(x) &= \frac{1}{x}\\
F(x) &= \int^a_b\frac{1}{3}x^3
y &= \sqrt{a}\\
z &= \sqrt[n]{a}
\end{align}
```

B. **Output:**

$$\begin{aligned}
x + y &= 5 && (2)\\
g(x) &= \frac{1}{x} && (3)\\
F(x) &= \int_b^a \frac{1}{3}x^3 && (4)\\
y &= \sqrt{a} && (5)\\
z &= \sqrt[n]{a} && (6)
\end{aligned}$$

ii. This will display the equation and number it.If you want to avoid numbering the equation, use the syntax -

A. **Syntax:**

```
\begin{align*}
x + y &= 5 \\
g(x) &= \frac{1}{x}\\
F(x) &= \int^a_b\frac{1}{3}x^3
y &= \sqrt{a}\\
z &= \sqrt[n]{a}
\end{align*}
```

B. **Output:**

$$\begin{aligned}
x + y &= 5\\
g(x) &= \frac{1}{x}\\
F(x) &= \int_b^a \frac{1}{3}x^3\\
y &= \sqrt{a}\\
z &= \sqrt[n]{a}
\end{aligned}$$

(e) **Matrix:**

To create matrices, you can use the bmatrix (for brackets around the matrix), pmatrix (for parentheses around the matrix), or matrix environment.

i. **Syntax:**

```
$\begin{matrix}
    a & b \\
    c & d
\end{matrix}$
\\
$\begin{bmatrix}
a & b \\
c & d
\end{bmatrix}$
\\
$\begin{pmatrix}
 a & b \\
 c & d
\end{pmatrix}$
```

ii. **Output:**

$$\begin{matrix} a & b \\ c & d \end{matrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

(f) **Table:**
Tables can be created using the tabular environment. Here's an example of a simple table:

i. **Syntax:**

```
\begin{tabular}{|c|c|}
\hline
Element & Value \\
\hline
1 & 10 \\
\hline
2 & 20 \\
\hline
3 & 30 \\
\hline
\end{tabular}
```

ii. **Output:**

Element	Value
1	10
2	20
3	30

V. Button Modification

Chin Tapak Dum Dum!

By Sunit Modak

This document describes the Java Swing application named `SymbolApp`. The application showcases a simple "mind-reading" trick by displaying a grid of symbols and revealing a selected symbol based on user interaction. This document is formatted using LaTeX to provide a clear and professional presentation for academic purposes.

1. Clone the Repository

At first, I used GitHub Desktop to clone the repository: https://github.com/GeekAyan/STT. Then I opened GitHub Desktop, clicked on "File" then "Clone Repository", pasted the URL, and selected my local directory.

2. Set Up the Project

I opened the project in VSCode. Then followed the detailed run instructions provided in the README.md file to set up any necessary dependencies and configurations. This could involve installing Python packages or setting environment variables.

3. Run the Application

I ran the application according to the instructions to ensure everything was working as expected.

4. Modify the Button

I located the code for the button in the project files. This could be in a JavaScript, HTML, or Python file, depending on the technology stack used. I renamed the button text to **Chin Tapak Dum Dum**.

5. Fix the Button Proportions

After renaming the button, I analyzed why the button looked disproportionate. Possible fixes could involve adjustments, and I modified the code accordingly.

6. Test the Changes

I ran the application again to ensure the button now appeared correctly proportioned and that it functioned as intended.

7. Commit the Changes

I saved all the changes in my IDE. In GitHub Desktop, I committed the changes with a descriptive message like “Fixed button proportions and renamed to ’Chin Tapak Dum Dum’”.

8. Push Changes

I pushed the changes to my forked version first.

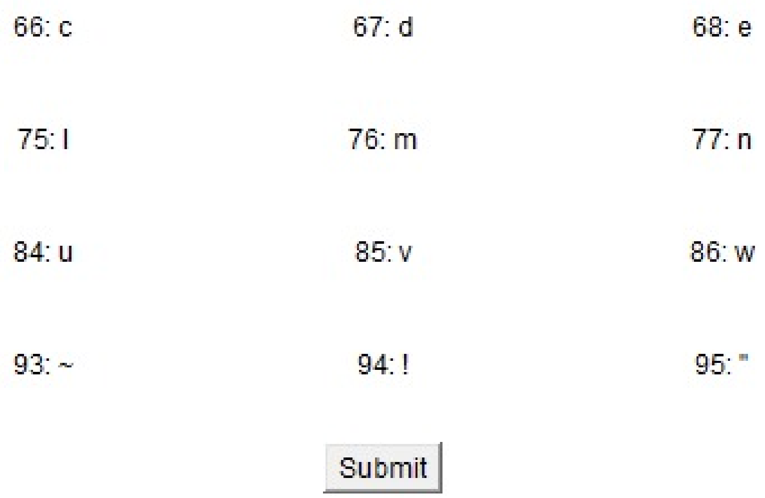
9. Create a Pull Request

I went to the original GitHub repository in my web browser. Clicked on “Pull Requests” then “New Pull Request”. Compared my branch with the main branch of the original repository. Added a title and description explaining my changes and why they were made.

Code Description

The `SymbolApp` class extends `Frame` and implements `ActionListener`. It generates a random symbol and displays it among other symbols in a grid layout. The user follows specific steps to select a symbol, which is then revealed when the submit button is clicked

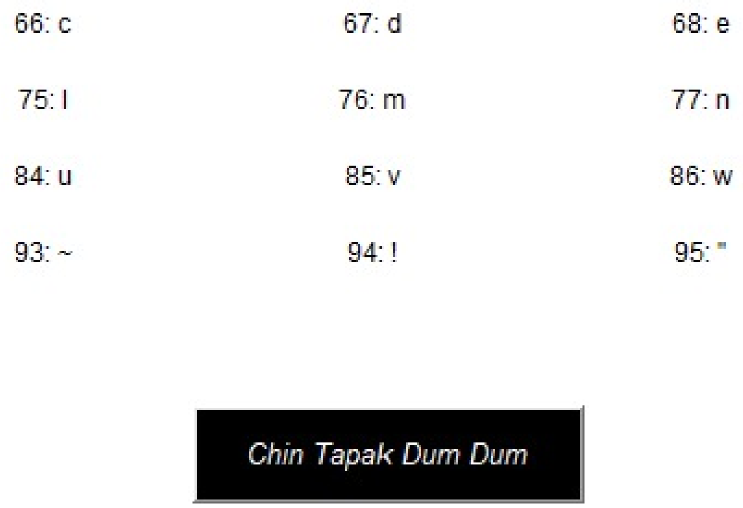
What we achieved! :



66: c	67: d	68: e
75: l	76: m	77: n
84: u	85: v	86: w
93: ~	94: !	95: "

Figure 3: Before

This is the original button which looks very ordinary and dull.



66: c	67: d	68: e
75: l	76: m	77: n
84: u	85: v	86: w
93: ~	94: !	95: "

Figure 4: After

This is the button after modifications, with a background color, improved proportions, and enhanced font. Looks nice, no?

Acknowledgement

We would like to express our deepest gratitude to our professor, **Mr. Ayan Ghosh**, for their invaluable guidance, encouragement, and support throughout this project. Their insightful feedback and constant availability helped us overcome many challenges, and their expertise provided direction for completing this work successfully.

We would also like to extend our sincere thanks to all the faculty members and MAKAUT, for their continuous support and for creating a positive environment conducive to learning and exploration. Additionally, we are immensely grateful to our fellow students and project group members for their collaborative efforts, cooperation, and dedication, without which this project would not have been possible. Finally, we are grateful to our families and friends for their understanding and moral support throughout the course of this work.

Signature of Student
(Name of Student)

Signature of Professor