
ASSIGNMENT NO:- 01

LATEX CODE:

```
\documentclass{article}

\title{History of LaTeX}
\author{Deepak Zumbar Shitole}
\date{\today}

\begin{document}

\maketitle
```

```
\section{Introduction}
```

LaTeX often stylized as LATEX) is a software system for document preparation.[3] When writing, the writer uses plain text as opposed to the formatted text found in WYSIWYG word processors like Microsoft Word, LibreOffice Writer and Apple Pages. The writer uses markup tagging conventions to define the general structure of a document, to style text throughout a document (such as bold and italics), and to add citations and cross-references. A TeX distribution such as TeX Live or MiKTeX is used to produce an output file (such as PDF or DVI) suitable for printing or digital distribution.\newline

LaTeX is widely used in academia[4][5] for the communication and publication of scientific documents in many fields, including mathematics, computer science, engineering, physics, chemistry, economics, linguistics, quantitative psychology, philosophy, and political science. It also has a prominent role in the preparation and publication of books and articles that contain complex multilingual materials, such as Arabic and Greek.[6] LaTeX uses the TeX typesetting program for formatting its output, and is itself written in the TeX macro language.

```
\section{History}
```

LaTeX was created in the early 1980s by Leslie Lamport when he was working at SRI. He needed to write TeX macros for his own use and thought that with a little extra effort, he could make a general package usable by others. Peter Gordon, an editor at Addison-Wesley, convinced him to write a LaTeX user's manual for publication (Lamport was initially skeptical that anyone would pay money for it);[12] it came out in 1986[3] and sold hundreds of thousands of copies.[12] Meanwhile, Lamport released versions of his LaTeX macros in 1984 and 1985. On 21 August 1989, at a TeX Users Group (TUG) meeting at Stanford, Lamport agreed to turn over maintenance and development of LaTeX to Frank Mittelbach. Frank Mittelbach, along with Chris Rowley and Rainer Schöpf, formed the LaTeX3 team; in 1994.

```
\section{Licensing}
```

LaTeX is typically distributed along with plain TeX under a free software license: the LaTeX Project Public License (LPPL).[36] The LPPL is not compatible with the GNU General Public License, as it requires that modified files must be clearly differentiable from their originals (usually by changing the filename); this was done to ensure that files that depend on other files will produce

the expected behavior and avoid dependency hell. The LPPL is DFSG compliant as of version 1.3. As free software, LaTeX is available on most operating systems, which include UNIX (Solaris, HP-UX, AIX), BSD (FreeBSD, macOS, NetBSD, OpenBSD), Linux (Red Hat, Debian, Arch, Gentoo), Windows, DOS, RISC OS, AmigaOS and Plan 9.

`\section{Versions}`

LaTeX2e is the current version of LaTeX, since it replaced LaTeX 2.09 in 1994.[37] As of 2020, LaTeX3, which started in the early 1990s, is under a long-term development project.[10] Planned features include improved syntax (separation of content from styling), hyperlink support, a new user interface, access to arbitrary fonts and a new documentation.[38] Some LaTeX3 features are available in LaTeX2e using packages,[39] and by 2020 many features have been enabled in LaTeX2e by default for a gradual transition.[10]\newline

There are numerous commercial implementations of the entire TeX system. System vendors may add extra features like additional typefaces and telephone support. LyX is a free, WYSIWYM visual document processor that uses LaTeX for a back-end.[40] TeXmacs is a free, WYSIWYG editor with similar functionalities as LaTeX, but with a different typesetting engine.[41] Other WYSIWYG editors that produce LaTeX include Scientific Word on Windows, and BaKoMa TeX on Windows, Mac and Linux.

`\end{document}`

OUTPUT:

History of LaTeX

Deepak Zumbar Shitole

December 12, 2023

1 Introduction

LaTeX (often stylized as L^AT_EX) is a software system for document preparation.[3] When writing, the writer uses plain text as opposed to the formatted text found in WYSIWYG word processors like Microsoft Word, LibreOffice Writer and Apple Pages. The writer uses markup tagging conventions to define the general structure of a document, to stylise text throughout a document (such as bold and italics), and to add citations and cross-references. A TeX distribution such as TeX Live or MiKTeX is used to produce an output file (such as PDF or DVI) suitable for printing or digital distribution.

LaTeX is widely used in academia[4][5] for the communication and publication of scientific documents in many fields, including mathematics, computer science, engineering, physics, chemistry, economics, linguistics, quantitative psychology, philosophy, and political science. It also has a prominent role in the preparation and publication of books and articles that contain complex multilingual materials, such as Arabic and Greek.[6] LaTeX uses the TeX typesetting program for formatting its output, and is itself written in the TeX macro language.

2 History

LaTeX was created in the early 1980s by Leslie Lamport when he was working at SRI. He needed to write TeX macros for his own use and thought that with a little extra effort, he could make a general package usable by others. Peter Gordon, an editor at Addison-Wesley, convinced him to write a LaTeX user's manual for publication (Lamport was initially skeptical that anyone would pay money for it);[12] it came out in 1986[3] and sold hundreds of thousands of copies.[12] Meanwhile, Lamport released versions of his LaTeX macros in 1984 and 1985. On 21 August 1989, at a TeX Users Group (TUG) meeting at Stanford, Lamport agreed to turn over maintenance and development of LaTeX to Frank Mittelbach. Frank Mittelbach, along with Chris Rowley and Rainer Schöpf, formed the LaTeX3 team; in 1994.

3 Licensing

LaTeX is typically distributed along with plain TeX under a free software license: the LaTeX Project Public License (LPPL).[36] The LPPL is not compatible with the GNU General Public License, as it requires that modified files must be clearly differentiable from their originals (usually by changing the filename); this was done to ensure that files that depend on other files will produce the expected behavior and avoid dependency hell. The LPPL is DFSG compliant as of version 1.3. As free software, LaTeX is available on most operating systems, which include UNIX (Solaris, HP-UX, AIX), BSD (FreeBSD, macOS, NetBSD, OpenBSD), Linux (Red Hat, Debian, Arch, Gentoo), Windows, DOS, RISC OS, AmigaOS and Plan 9.

4 Versions

LaTeX2e is the current version of LaTeX, since it replaced LaTeX 2.09 in 1994.[37] As of 2020, LaTeX3, which started in the early 1990s, is under a long-term development project.[10] Planned features include improved syntax (separation of content from styling), hyperlink support, a new user interface, access to arbitrary fonts and a new documentation.[38] Some LaTeX3 features are available in LaTeX2e using packages,[39] and by 2020 many features have been enabled in LaTeX2e by default for a gradual transition.[10]

There are numerous commercial implementations of the entire TeX system. System vendors may add extra features like additional typefaces and telephone support. LyX is a free, WYSIWYM visual document processor that uses LaTeX for a back-end.[40] TeXmacs is a free, WYSIWYG editor with similar functionalities as LaTeX, but with a different typesetting engine.[41] Other WYSIWYG editors that produce LaTeX include Scientific Word on Windows, and BaKoMa TeX on Windows, Mac and Linux.

ASSIGNMENT NO:-02

LATEX CODE:

```
\documentclass{IEEEtran}
\usepackage{amsmath}
\usepackage{cite}
\usepackage{graphicx}

\title{Fluid Structure Interaction : A Brief Review}
\author{Chandrasekhar Jinendran} % \\ for new line
\date{\today}

\begin{document}
\maketitle

\section{Overview}
```

Section \ref{sec:intro} shows an outline of the literature. Sections \ref{subsec:way} and \ref{subsec:approach} on page \pageref{subsec:way} covers the common methodologies in solving FSI problems. Section \ref{subsec:coupling} refers to the different methods of partitioned approach and section \ref{sec:disc} explains about the most common discretisation method in FSI named ALE(Arbitrary Lagrangian Eulerian) approach. Section \ref{sec:eqn} refers to the governing equations of FSI. Finally, section \ref{sec:bc} explains the boundary conditions imposed generally to an FSI problem.

% ch: for chapter; fig: for figure; tbl: for table

```
\section{Introduction}
\label{sec:intro}
```

A system is said to be coupled when two or more physical systems are interacting together and solution of each system is dependent on the solution of the other. An example of a coupled system is Fluid Structure Interaction where there is interaction between a fluid and a structure. Here neither the fluid part nor the structural part can be solved independently because of the unknown forces in the interface region. In fluid structure interaction, the solid and fluid systems exchange mechanical energy at the interface in both directions. The fluid exerts forces on the moving solid changing its dynamics and the structure causes displacements in the fluid changing the characteristics of flow. In many applications, the forces from the moving fluid deforms the structure only by small amount. In those cases the response from the structure can be solved independently when fluid flow characteristics are determined, since the deformation does not affect the flow. Systems needed to be coupled when the fluid forces causes considerable deformation of the structure causing change in the fluid pattern.

Some of the complications arise from the fact that fluid and solid mesh systems are quite different. Frequently, the fluid system uses an Eulerian or spatially fixed-coordinate system, while the solid system uses a Lagrangian or material fixed-coordinate system. Hence, we must take care to develop a suitable interfacing technique between the two modules. Also,

the time scales can be different for the two modules; hence one must be careful while performing coupled calculations.

\section{Theory}

\subsection{One-Way and Two-Way FSI}

\label{subsec:way}

In One-Way FSI, either the fluid or structural domain is initially solved independently, and the results are employed as a model condition when solving the other domain. In a Two-Way FSI study, the fluid and structural domains are solved in parallel. At each sub-step the fluid and structural domain solutions are required to converge before moving to the next step\cite{b1}. The fluid and structural domains are connected through a coupling scheme, like the Arbitrary Lagrangian Euler method, where the mesh of one domain is allowed to move so that it adheres to the boundaries of the other domain, or IB method, where the mesh does not deform and structural movement within the fluid is accounted for by adding body forces to the equations of motion. Fully-coupled Two-Way FSI studies are typically transient and capture fluid structure behavior over a period.

\subsection{Monolithic and Partitioned Approach}

\label{subsec:approach}

Generally FSI problems are too complex to solve analytically so their numerical simulations are very important. The two main approaches are monolithic and partitioned approach. In monolithic or direct approach, the governing equations of fluid and structural displacements are solved simultaneously. The discretisation of the problem results in a large number of equations which is solved with a single solver. This approach has the advantage of stability since the mutual influence of the solid and fluid can be taken into account directly. Whereas in partitioned approach, governing equations of the fluid and solid domains are solved separately with two separate solvers\cite{b2}. Here a coupling algorithm is required for interaction and for solving the coupled problem. The basic procedure of partitioned approach consists of exchange of data between fluid and structural solvers alternatively at the interface boundary for every instant. It is easy to implement due to the freedom in choosing fluid and structural solvers.

```
\begin{figure}[h]
\centering
\includegraphics [width=0.3\textwidth] {graph.png}
\caption{This is an example image.}
\label{fig:example}
\end{figure}
```

Since a monolithic coupling procedure considers all the fluid and structure problem unknowns simultaneously at the same instant, the complete system becomes difficult to solve. So a partitioned coupling is preferred for real-world applications.

\subsection{Loosely and Strongly Coupled}

\label{subsec:coupling}

In partitioned approach, interaction can be either loosely or strongly coupled. A loosely coupled algorithm is explicit and there will be bidirectional exchange of solved variables per time step sequentially. For loosely coupled algorithms stability is dependent on mass density ratio (ratio of solid density to fluid density) and geometry of the domains. Here stability decreases with decreasing mass density ratio. For unstable computations, decreasing of time step size will lead to earlier occurrence of instabilities or increased instabilities. For conditionally stable computations the time step size has an upper limit depending on the CFL number (Courant-Friedrichs-Lewy condition) and a lower limit depending on the highest eigen mode of the added mass matrix (since fluid closest to the coupling interface will act as extra mass on the structure).

Whereas a strongly coupled algorithm is implicit and performs iterations to converge at the end of time step. They are more stable for low mass density ratio than the loosely coupled algorithm.

Decrease of this ratio leads to more sub-iterations which in turn leads to increased computing time.

\section{Discretisation}

\label{sec:disc}

To account for the deformation of the mesh, ALE formulation of the Navier Stokes equations is used, discretized on a control volume with volume $V(t)$ and boundary $S(t)$. ALE is used in order to combine the advantages of the Lagrangian and Eulerian formulations, as they are usually exploited for individual structural and fluid mechanics approaches, respectively. The principal idea of the ALE approach is that an observer is neither located at a fixed position in space nor does it move with the material point, but can move arbitrarily\cite{b3}.

\section{Equations}

\label{sec:eqn}

The FSI problem consist of a fluid that is occupying a given domain Ω^F and a structure that occupies another domain Ω^S which interact at the common boundary Γ \cite{b4}.

\subsection{Structural Equation}

The large displacement of the structure is governed by:

\begin{equation}

$$\rho^S \frac{D^2 u}{D t^2} - \nabla \cdot (F \cdot S(u)) = \rho^S b^S \text{ in } \Omega^S$$

\end{equation}

where u represents the displacements of the structure, b^S represents the body forces applied on the structure. S represents the second Piola-Kirchhoff stress tensor, ρ^S represents the density of the structure and F represents the deformation gradient tensor.

\subsection{Fluid Flow Equation}

The fluid equations to be solved are the incompressible Navier-Stokes equations expressed in ALE formulation. The Navier Stokes equations can be derived from the conservation laws of mass and linear momentum and takes into consideration that the fluid is viscous. The equations in ALE formulation take the form:

$$\rho \frac{dv}{dt} = \rho b + \nabla \cdot \tau + \nabla p \quad \text{in } \Omega$$

$$\nabla \cdot v = 0 \quad \text{in } \Omega$$

Here v denotes the fluid velocity and p denotes the physical pressure. The fluid density and viscosity is given by ρ and μ respectively. The fluid body forces are represented by b and τ represents the strain rate tensor. As can be seen in (2) the ALE formulation comes into the equation in the fluid acceleration term and the convective term.

\subsection{Coupling Equation}

At the interface Γ , kinematic and dynamic continuity is required. The governing kinematic coupling equation is:

$$u_\Gamma(t) = d_\Gamma(t)$$

Here $d_\Gamma(t)$ represents the displacement of the fluid mesh nodes at the interface.

The dynamic coupling equation takes the form:

$$h^S(t) + h^F(t) = 0$$

where $h = \sigma \cdot n$ signifies the traction vector.

\section{Boundary Conditions}

\label{sec:bc}

Regarding the boundaries in FSI, along solid-wall boundaries, the particle velocity is coupled to the rigid or flexible structure. The kinematic requirement is that no particle can cross the interface. Due to the coupling between fluid and structure, one also needs to define some coupling conditions so the fluid and the structure do not overlap or detach during the

motion\cite{b5}. These conditions depend on the fluid and also differ depending on whether the fluid is inviscid or viscous.

```
\begin{thebibliography}{00}
\bibitem{b1} R Ilangoan, K Nagamani and P Gopal Swamy, 2007, Recycling of Quarry
Waste as
an Alternative Material in Concrete Manufacturing, Indian Construction, vol 40, no 2
\bibitem{b2} Ilangoan R and Nagamani K. Application of quarry Rock dust as fine
aggregate in
concrete construction. National Journal on construction Management: NICMR. Pune.
\bibitem{b3} A K Sahu, S Kumar and A K Sachin.,2003 Crushed Stone Waste as Fine
Aggregate
for Concrete.The Indian Concrete Journal, vol 77, no 1, p 845.
\bibitem{b4} D S Prakash Rao and V Giridhar Kumar., 2004, Investigationon Concrete With
Stone
Crusher Dust as Fine Aggregate.The Indian Concrete Journal, vol 78, no 7, p 45.
\bibitem{b5} Ahmed E. Ahmed , Ahmed A. El-Kour, 1989, Properties of concrete
incorporating natural and crushed stone very fine sand, ACIV Material journal, Vol-86, No. 4.
\bibitem{b6} M R Chitlange,, Appraisal of Artificial Sand Concrete, IE(I)Journal Volume90
\bibitem{b7} Code of Practice for Plain and Reinforced Concrete IS 456: 2000, Bureau of
Indian Standards, New Delhi.
\end{thebibliography}

\end{document}
```

OUTPUT:

1

Fluid Structure Interaction : A Brief Review

Chandrasekhar Jinendran

I. OVERVIEW

Section II shows an outline of the literature. Sections III-A and III-B on page 1 covers the common methodologies in solving FSI problems. Section III-C refers to the different methods of partitioned approach and section IV explains about the most common discretisation method in FSI named ALE(Arbitrary Lagrangian Eulerian) approach. Section V refers to the governing equations of FSI. Finally, section VI explains the boundary conditions imposed generally to an FSI problem.

II. INTRODUCTION

A system is said to be coupled when two or more physical systems are interacting together and solution of each system is dependent on the solution of the other. An example of a coupled system is Fluid Structure Interaction where there is interaction between a fluid and a structure. Here neither the fluid part nor the structural part can be solved independently because of the unknown forces in the interface region. In fluid structure interaction, the solid and fluid systems exchange mechanical energy at the interface in both directions. The fluid exerts forces on the moving solid changing its dynamics and the structure causes displacements in the fluid changing the characteristics of flow. In many applications, the forces from the moving fluid deforms the structure only by small amount. In those cases the response from the structure can be solved independently when fluid flow characteristics are determined, since the deformation does not affect the flow. Systems needed to be coupled when the fluid forces causes considerable deformation of the structure causing change in the fluid pattern.

Some of the complications arise from the fact that fluid and solid mesh systems are quite different. Frequently, the fluid system uses an Eulerian or spatially fixed-coordinate system, while the solid system uses a Lagrangian or material fixed-coordinate system. Hence, we must take care to develop a suitable interfacing technique between the two modules. Also, the time scales can be different for the two modules; hence one must be careful while performing coupled calculations.

III. THEORY

A. One-Way and Two-Way FSI

In One-Way FSI, either the fluid or structural domain is initially solved independently, and the results are employed as a model condition when solving the other domain. In a Two-Way FSI study, the fluid and structural domains are solved in parallel. At each sub-step the fluid and structural domain solutions are required to converge before moving to the next step [1]. The fluid and structural domains are connected through a coupling scheme, like the Arbitrary Lagrangian Euler method, where the mesh of one domain is allowed to

move so that it adheres to the boundaries of the other domain, or IB method, where the mesh does not deform and structural movement within the fluid is accounted for by adding body forces to the equations of motion. Fully-coupled Two-Way FSI studies are typically transient and capture fluid structure behavior over a period.

B. Monolithic and Partitioned Approach

Generally FSI problems are too complex to solve analytically so their numerical simulations are very important. The two main approaches are monolithic and partitioned approach. In monolithic or direct approach, the governing equations of fluid and structural displacements are solved simultaneously. The discretisation of the problem results in a large number of equations which is solved with a single solver. This approach has the advantage of stability since the mutual influence of the solid and fluid can be taken into account directly. Whereas in partitioned approach, governing equations of the fluid and solid domains are solved separately with two separate solvers [2]. Here a coupling algorithm is required for interaction and for solving the coupled problem. The basic procedure of partitioned approach consists of exchange of data between fluid and structural solvers alternatively at the interface boundary for every instant. It is easy to implement due to the freedom in choosing fluid and structural solvers.



For conditionally stable computations the time step size has an upper limit depending on the CFL number (Courant-Friedrichs-Lewy condition) and a lower limit depending on the highest eigen mode of the added mass matrix (since fluid closest to the coupling interface will act as extra mass on the structure).

Whereas a strongly coupled algorithm is implicit and performs iterations to converge at the end of time step. They are more stable for low mass density ratio than the loosely coupled algorithm. Decrease of this ratio leads to more sub-iterations which in turn leads to increased computing time.

IV. DISCRETISATION

To account for the deformation of the mesh, ALE formulation of the Navier Stokes equations is used, discretized on a control volume with volume $V(t)$ and boundary $S(t)$. ALE is used in order to combine the advantages of the Lagrangian and Eulerian formulations, as they are usually exploited for individual structural and fluid mechanics approaches, respectively. The principal idea of the ALE approach is that an observer is neither located at a fixed position in space nor does it move with the material point, but can move arbitrarily [3].

V. EQUATIONS

The FSI problem consist of a fluid that is occupying a given domain Ω^F and a structure that occupies another domain Ω^S which interact at the common boundary τ [4].

A. Structural Equation

The large displacement of the structure is governed by:

$$\rho^S \frac{D^2 u}{Dt^2} - \nabla \cdot (F \cdot S(u)) = \rho^S b^S \text{ in } \Omega^S \quad (1)$$

where u represents the displacements of the structure, b^S represents the body forces applied on the structure. S represents the second Piola-Kirchhoff stress tensor, ρ^S represents the density of the structure and F represents the deformation gradient tensor.

B. Fluid Flow Equation

The fluid equations to be solved are the incompressible Navier-Stokes equations expressed in ALE formulation. The Navier Stokes equations can be derived from the conservation laws of mass and linear momentum and takes into consideration that the fluid is viscous. The equations in ALE formulation take the form:

$$\rho^F \frac{dv}{dt} \Big|_x + \rho^F \cdot \nabla v - 2 \mu \nabla \cdot \varepsilon(v) + \nabla p = \rho^F b^F \text{ in } \Omega^F \quad (2)$$

$$\nabla v = 0 \text{ in } \Omega^F \quad (3)$$

Here v denotes the fluid velocity and p denotes the physical pressure. The fluid density and viscosity is given by ρ^F and μ respectively. The fluid body forces are represented by b^F and $\varepsilon(v)$ represents the strain rate tensor. As can be seen in (2) the ALE formulation comes into the equation in the fluid acceleration term and the convective term.

C. Coupling Equation

At the interface τ , kinematic and dynamic continuity is required. The governing kinematic coupling equation is:

$$u_\tau(t) = d_\tau^F(t) \quad (4)$$

Here $d_\tau^F(t)$ represents the displacement of the fluid mesh nodes at the interface. The dynamic coupling equation takes the form:

$$h^S(t) + h^F(t) = 0 \quad (5)$$

where $h = \sigma \cdot n$ signifies the traction vector.

VI. BOUNDARY CONDITIONS

Regarding the boundaries in FSI, along solid-wall boundaries, the particle velocity is coupled to the rigid or flexible structure. The kinematic requirement is that no particle can cross the interface. Due to the coupling between fluid and structure, one also needs to define some coupling conditions so the fluid and the structure do not overlap or detach during the motion [5]. These conditions depend on the fluid and also differ depending on whether the fluid is inviscid or viscous.

REFERENCES

- [1] R Ilangoan, K Nagamani and P Gopal Swamy, 2007, Recycling of Quarry Waste as an Alternative Material in Concrete Manufacturing, Indian Construction, vol 40, no 2
- [2] Ilangoan R and Nagamani K, Application of quarry Rock dust as fine aggregate in concrete construction. National Journal on construction Management: NICMR, Pune.
- [3] A K Sahu, S Kumar and A K Sachin, 2003 Crushed Stone Waste as Fine Aggregate for Concrete. The Indian Concrete Journal, vol 77, no 1, p 845.
- [4] D S Prakash Rao and V Giridhar Kumar, 2004, Investigation on Concrete With Stone Crusher Dust as Fine Aggregate. The Indian Concrete Journal, vol 78, no 7, p 45.
- [5] Ahmed E. Ahmed, Ahmed A. El-Koud, 1989, Properties of concrete incorporating natural and crushed stone very fine sand, ACIV Material journal, Vol-86, No. 4.
- [6] M R Chitlange, Appraisal of Artificial Sand Concrete, IE(I)Journal Volume 90
- [7] Code of Practice for Plain and Reinforced Concrete IS 456: 2000, Bureau of Indian Standards, New Delhi.

ASSIGNMENT NO:-03

```
Open  [icon]  assign3.sh
~/Desktop/deepak dtl

1 #!/bin/bash
2
3 sum=0
4 count=0
5
6 while true; do
7     echo -n "Enter a number (or any non-numeric value to finish): "
8     read num
9
10    if [[ $num =~ ^[0-9]+$ ]]; then
11        sum=$((sum + num))
12        count=$((count + 1))
13    else
14        break
15    fi
16 done
17
18 if [ $count -gt 0 ]; then
19     average=$((echo "scale=2; $sum / $count" | bc))
20     echo "Sum: $sum"
21     echo "Average: $average"
22 else
23     echo "No valid numbers entered. Exiting."
24 fi
25
```

```
local@ubuntu1:~/Desktop/deepak dtl$ ./assign3.sh
Enter a number (or any non-numeric value to finish): 22
Enter a number (or any non-numeric value to finish): 9
Enter a number (or any non-numeric value to finish): 2003
Enter a number (or any non-numeric value to finish): d
Sum: 2034
Average: 678.00
local@ubuntu1:~/Desktop/deepak dtl$
```

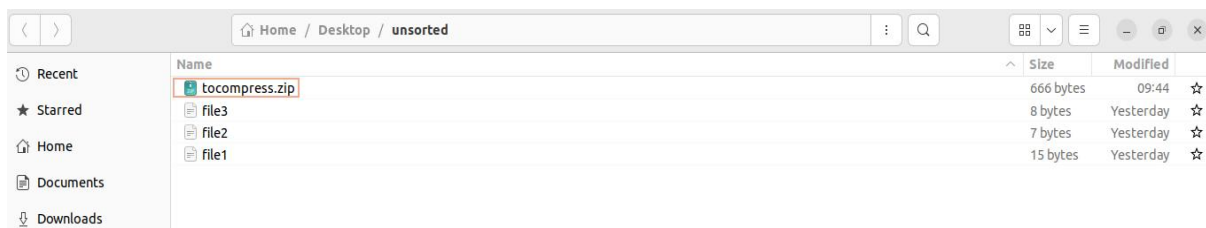
NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

ASSIGNMENT NO:-04

```
Open [x] assign4.sh ~/Desktop/deepak dtl Save [≡] [−] [⌂] [X]
1 #!/bin/bash
2
3 echo -n "Enter the path to the compressed file: "
4 read compressed_file
5
6 if [ ! -f "$compressed_file" ]; then
7     echo "File not found. Exiting."
8     exit 1
9 fi
10
11 file_type=$(file -b --mime-type "$compressed_file")
12
13 case "$file_type" in
14     "application/zip") unzip "$compressed_file" ;;
15     "application/x-tar" | "application/gzip" | "application/x-gzip" | "application/x-bzip2" | "application/x-xz") tar -xf "$compressed_file" ;;
16     "application/x-rar-compressed") unrar x "$compressed_file" ;;
17     "application/x-7z-compressed") 7z x "$compressed_file" ;;
18     *) echo "Unsupported compression format. Exiting." ;;
19 esac
20
```

```
local@ubuntu1:~/Desktop/deepak dtl$ ./assign4.sh
Enter the path to the compressed file: /home/local/Desktop/unsorted/tocompress.zip
Archive: /home/local/Desktop/unsorted/tocompress.zip
  creating: unsorted/
  extracting: unsorted/file1
  extracting: unsorted/file2
  extracting: unsorted/file3
local@ubuntu1:~/Desktop/deepak dtl$
```



Home / Desktop / unsorted		
Name	Size	Modified
tocompress.zip	666 bytes	09:44 ☆
file3	8 bytes	Yesterday ☆
file2	7 bytes	Yesterday ☆
file1	15 bytes	Yesterday ☆

NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

ASSIGNMENT NO:-05

```
Open  [icon] assign5.sh
~/Desktop/deepak dtl

1#!/bin/bash
2
3echo -n "Enter the path to the CSV file: "
4read csv_file
5
6if [ ! -f "$csv_file" ]; then
7    echo "File not found. Exiting."
8    exit 1
9fi
10
11vcf_file="contacts.vcf"
12echo "" > "$vcf_file"
13
14while IFS=',' read -r name phone email; do
15    if [[ "$name" == "Name" && "$phone" == "Phone" && "$email" == "Email" ]]; then
16        continue
17    fi
18
19    echo "BEGIN:VCARD" >> "$vcf_file"
20    echo "VERSION:3.0" >> "$vcf_file"
21    echo "FN:$name" >> "$vcf_file"
22    echo "TEL:$phone" >> "$vcf_file"
23    echo "EMAIL:$email" >> "$vcf_file"
24    echo "END:VCARD" >> "$vcf_file"
25done < "$csv_file"
26
27echo "Conversion complete. VCF file saved as $vcf_file."
28
```

```
local@ubuntu1:~/Desktop/deepak dtl$ ./assign5.sh
Enter the path to the CSV file: /home/local/Desktop/unsorted/test.csv
Conversion complete. VCF file saved as contacts.vcf.
local@ubuntu1:~/Desktop/deepak dtl$
```

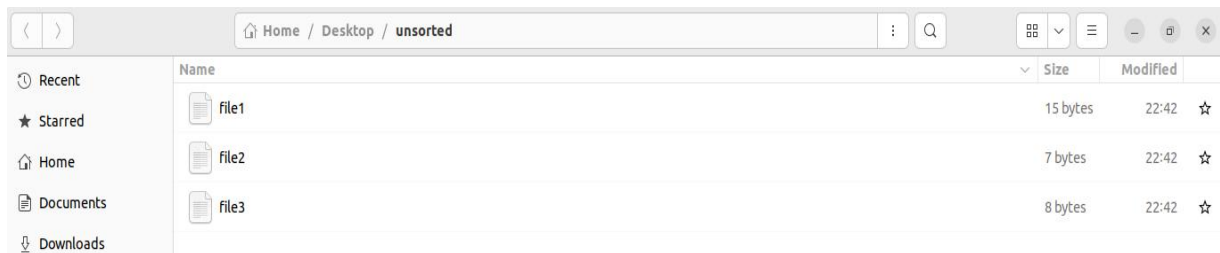
Name	Size	Modified	
unsorted	3 items	Yesterday	☆
assign3.sh	434 bytes	09:38	☆
assign4.sh	634 bytes	09:46	☆
assign5.sh	651 bytes	09:54	☆
assign6.sh	250 bytes	Yesterday	☆
assign7.sh	1.5 kB	Yesterday	☆
assign8.sh	459 bytes	10:19	☆
contacts.vcf	114 bytes	14:19	☆
todo.txt	36 bytes	Yesterday	☆

ASSIGNMENT NO:-06

PROGRAM:

```
Open  [icon] assign6.sh
~/Desktop/deepak dtl
1 #!/bin/bash
2
3 if [ -z "$1" ]; then
4     echo "Usage: $0 <directory>"
5     exit 1
6 fi
7
8 directory="$1"
9
10 if [ ! -d "$directory" ]; then
11     echo "Error: '$directory' is not a directory."
12     exit 1
13 fi
14
15 find "$directory" -type f -exec du -h {} + | sort -rh
16
```

OUTPUT:



Name	Size	Modified
file1	15 bytes	22:42 ☆
file2	7 bytes	22:42 ☆
file3	8 bytes	22:42 ☆

```
local@ubuntu1:~/Desktop/deepak dtl$ ls
assign6.sh assign7.sh todo.txt
local@ubuntu1:~/Desktop/deepak dtl$ chmod +x assign6.sh
local@ubuntu1:~/Desktop/deepak dtl$ ./assign6.sh /home/local/Desktop/unsorted
4.0K    /home/local/Desktop/unsorted/file3
4.0K    /home/local/Desktop/unsorted/file2
4.0K    /home/local/Desktop/unsorted/file1
local@ubuntu1:~/Desktop/deepak dtl$
```

ASSIGNMENT NO:-07

PROGRAM:

```
Open ▾ [🔍] *assign7.sh
~/Desktop/deepak dtl

1 #!/bin/bash
2
3 TODO_FILE="todo.txt"
4
5 # Check if the todo file exists, create it if not
6 if [ ! -e "$TODO_FILE" ]; then
7     touch "$TODO_FILE"
8 fi
9
10 print_help() {
11     echo "Usage: $0 [option] [task]"
12     echo "Options:"
13     echo "  add <task>:      Add a new task"
14     echo "  remove <task>:   Remove a task"
15     echo "  list:           List all tasks"
16     echo "  sort:          Sort tasks alphabetically"
17     echo "  prepend <task>: Prepend a task"
18     echo "  append <task>:  Append a task"
19     echo "  deduplicate:    Remove duplicate tasks"
20 }
21
22 add_task() {
23     echo "$1" >> "$TODO_FILE"
24     echo "Task added: $1"
25 }
26
27
28 remove_task() {
29     sed -i "/$1/d" "$TODO_FILE"
30     echo "Task removed: $1"
31 }
32
33 list_tasks() {S
34     echo "To-Do List:"
35     cat -n "$TODO_FILE"
36 }
```

```
38 sort_tasks() {
39     sort -o "$TODO_FILE" "$TODO_FILE"
40     echo "Tasks sorted alphabetically"
41 }
42
43 prepend_task() {
44     echo "$1" | cat - "$TODO_FILE" > temp && mv temp "$TODO_FILE"
45     echo "Task prepended: $1"
46 }
47
48 append_task() {
49     echo "$1" >> "$TODO_FILE"
50     echo "Task appended: $1"
51 }
52
53 deduplicate_tasks() {
54     sort -u -o "$TODO_FILE" "$TODO_FILE"
55     echo "Duplicate tasks removed"
56 }
57
58 if [ "$#" -lt 1 ]; then
59     print_help
60     exit 1
61 fi
62
63 case "$1" in
64     "add") add_task "${*:2}" ;;
65     "remove") remove_task "${*:2}" ;;
66     "list") list_tasks ;;
67     "sort") sort_tasks ;;
68     "prepend") prepend_task "${*:2}" ;;
69     "append") append_task "${*:2}" ;;
70     "deduplicate") deduplicate_tasks ;;
```


OUTPUT:


```
local@ubuntu1:~/Desktop/deepak dtl$ ls
assign6.sh  assign7.sh  todo.txt
local@ubuntu1:~/Desktop/deepak dtl$ chmod +x assign7.sh
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh add "Reading"
Task added: Reading
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh add "Exercise"
Task added: Exercise
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh add "Study"
Task added: Study
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh add "Game"
Task added: Game
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh list
To-Do List:
1 coding
2 cooking
3 Reading
4 Exercise
5 Study
6 Game
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh sort
Tasks sorted alphabetically
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh list
To-Do List:
1 coding
2 cooking
3 Exercise
4 Game
5 Reading
6 Study
```

```
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh remove "coding"
Task removed: coding
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh list
To-Do List:
1 cooking
2 Exercise
3 Game
4 Reading
5 Study
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh prepend "Exercise"
Task prepended: Exercise
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh list
To-Do List:
1 Exercise
2 cooking
3 Exercise
4 Game
5 Reading
6 Study
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh deduplicate
Duplicate tasks removed
local@ubuntu1:~/Desktop/deepak dtl$ ./assign7.sh list
To-Do List:
1 cooking
2 Exercise
3 Game
4 Reading
5 Study
```


NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

ASSIGNMENT NO:-08

```
Open ▾  assign8.sh  
~/Desktop/deepak dtl  
1#!/bin/bash  
2  
3if [ "$#" -ne 3 ]; then  
4    echo "Usage: $0 <file_name> <pattern> <action>"  
5    exit 1  
6fi  
7  
8file_name=$1  
9pattern=$2  
10action=$3  
11  
12if [ ! -f "$file_name" ]; then  
13    echo "Error: File $file_name not found."  
14    exit 1  
15fi  
16  
17while IFS= read -r line; do  
18    fields=($line)  
19  
20    if [[ "$line" =~ $pattern ]]; then  
21        echo "Matched: $line"  
22    elif [[ "${fields[0]}" =~ $pattern ]]; then  
23        echo "Matched field: $line"  
24    fi  
25done < "$file_name"  
26
```

```
Open ▾  *names  
~/Desktop/unsorted  
1 Deepak 22  
2 Om 31  
3 Aman 12S
```

```
local@ubuntu1:~/Desktop/deepak dtl$ ./assign8.sh /home/local/Desktop/unsorted/names "22" "echo 'DOB : \${line}'"  
Matched: Deepak 22  
local@ubuntu1:~/Desktop/deepak dtl$
```

ASSIGNMENT NO:-09

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/local/Desktop/DTL/Assignments/Assignment9/.git/
```

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    calculator.c

nothing added to commit but untracked files present (use "git add" to track)
```

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git add calculator.c
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   calculator.c
```

NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   calculator.c

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   calculator.c

local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git add calculator.c
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git status
On branch master

No commits yet


Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   calculator.c
```

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git config --global user.email "204057.dzscwit@gmail.com"
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git config --global user.name "Deepak22903"
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git commit -m "Added exponent oprator to calculator program"
[master (root-commit) 21f3204] Added exponent oprator to calculator program
 1 file changed, 16 insertions(+)
 create mode 100644 calculator.c
```

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     double num1, num2;
6     printf("Enter two numbers: ");
7     scanf("%lf %lf", &num1, &num2);
8
9     // Add exponent operator
10    double result = pow(num1, num2);
11
12    printf("Result: %lf\n", result);
13
14    return 0;
15 }
16
```

NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

```
Open ▾   
1 #include <stdio.h>  
2 #include <math.h>  
3  
4 int main() {  
5     double num1, num2;  
6     printf("Enter two numbers: ");  
7     scanf("%lf %lf", &num1, &num2);  
8  
9     printf("Result: %lf\n", result);  
10  
11     return 0;  
12 }  
13 |
```

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment9$ git diff  
diff --git a/calculator.c b/calculator.c  
index d75d0b2..d89377e 100644  
--- a/calculator.c  
+++ b/calculator.c  
@@ -6,7 +6,7 @@ int main() {  
    printf("Enter two numbers: ");  
    scanf("%lf %lf", &num1, &num2);  
  
-    // Add exponent operator  
+    // Add exponent operatorrr  
    double result = pow(num1, num2);  
  
    printf("Result: %lf\n", result);  
}
```

```
Deepak Shitole@DeepakLaptop MINGW64 /d/B. Tech/DTL/Assignment9 (master)  
$ git push  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 12 threads  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 288 bytes | 288.00 KiB/s, done.  
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
To https://github.com/Deepak22903/DTL_Assignment_9.git  
1f74533..b4934c6 master -> master
```


NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

ASSIGNMENT NO:-10

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/local/Desktop/DTL/Assignments/Assignment10/.git/
```

```
Switched to branch 'master'
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git checkout master
Switched to branch 'master'
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git checkout -b decimal-calculator
Switched to a new branch 'decimal-calculator'
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git add calculator.c
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git commit -m "Added decimal addition function"
[decimal-calculator 4984bd8] Added decimal addition function
1 file changed, 4 insertions(+)
```

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git checkout -b hex-calculator
Switched to a new branch 'hex-calculator'
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git add calculator.c
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git commit -m "Added hexadecimal addition function"
[hex-calculator (root-commit) 7684790] Added hexadecimal addition function
1 file changed, 20 insertions(+)
create mode 100644 calculator.c
```

```
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git checkout master
Switched to branch 'master'
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git merge hex-calculator
Already up to date.
local@local-VirtualBox:~/Desktop/DTL/Assignments/Assignment10$ git merge decimal-calculator
Updating 7684790..4984bd8
Fast-forward
 calculator.c | 4 ++++
1 file changed, 4 insertions(+)
```


NAME: DEEPAK SHITOLE
MIS NO: 642303019

SUBJECT: DTL
ASSIGNMENTS

```
Deepak Shitole@DeepakLaptop MINGW64 /d/B. Tech/DTL/Assignment10 (master|MERGING)
$ git add calculator.c

Deepak Shitole@DeepakLaptop MINGW64 /d/B. Tech/DTL/Assignment10 (master|MERGING)
$ git commit -m "final commit"
[master 175f8fb] final commit
```

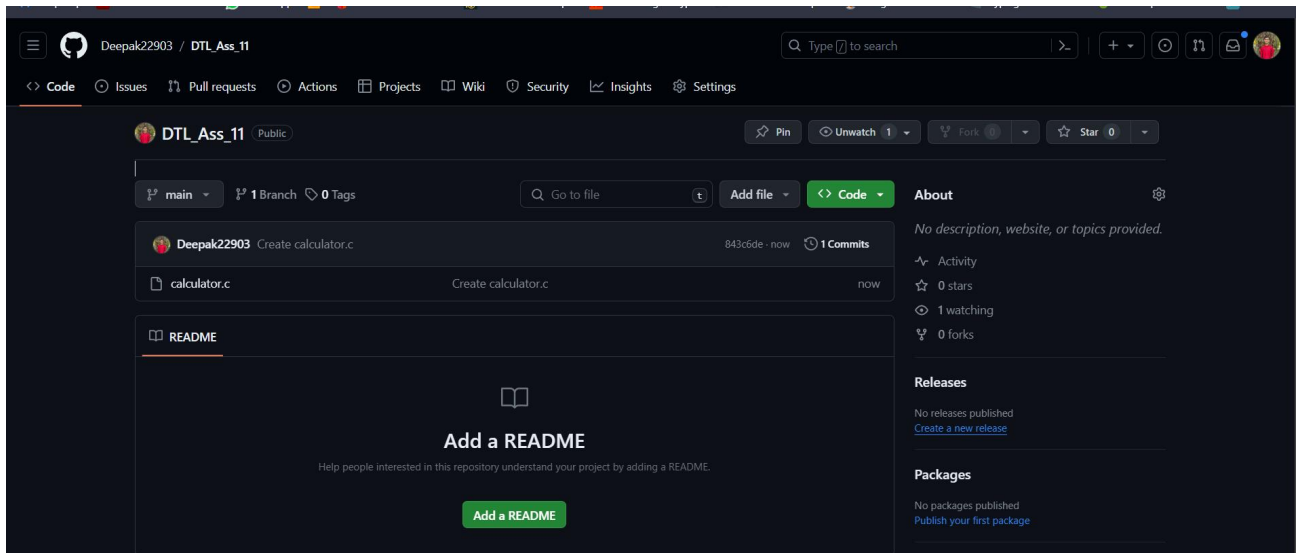
```
Deepak Shitole@DeepakLaptop MINGW64 /d/B. Tech/DTL/Assignment10 (master)
$ git push --set-upstream origin master
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 812 bytes | 812.00 KiB/s, done.
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To https://github.com/Deepak22903/DTL_Assignment_10.git
   1f74533..175f8fb  master -> master
branch 'master' set up to track 'origin/master'.

Deepak Shitole@DeepakLaptop MINGW64 /d/B. Tech/DTL/Assignment10 (master)
$

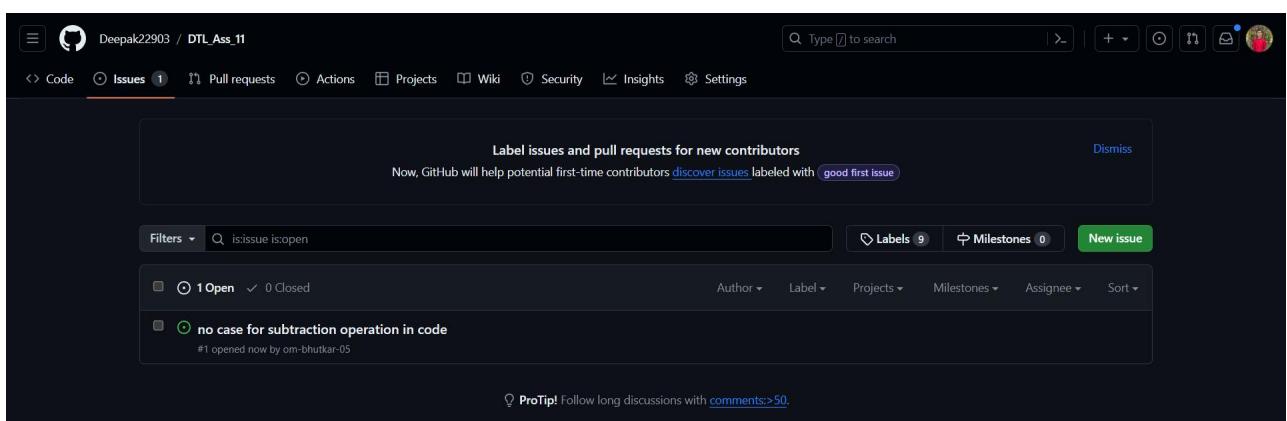
Deepak Shitole@DeepakLaptop MINGW64 /d/B. Tech/DTL/Assignment10 (master)
$ git push
Everything up-to-date
```

ASSIGNMENT NO:-11

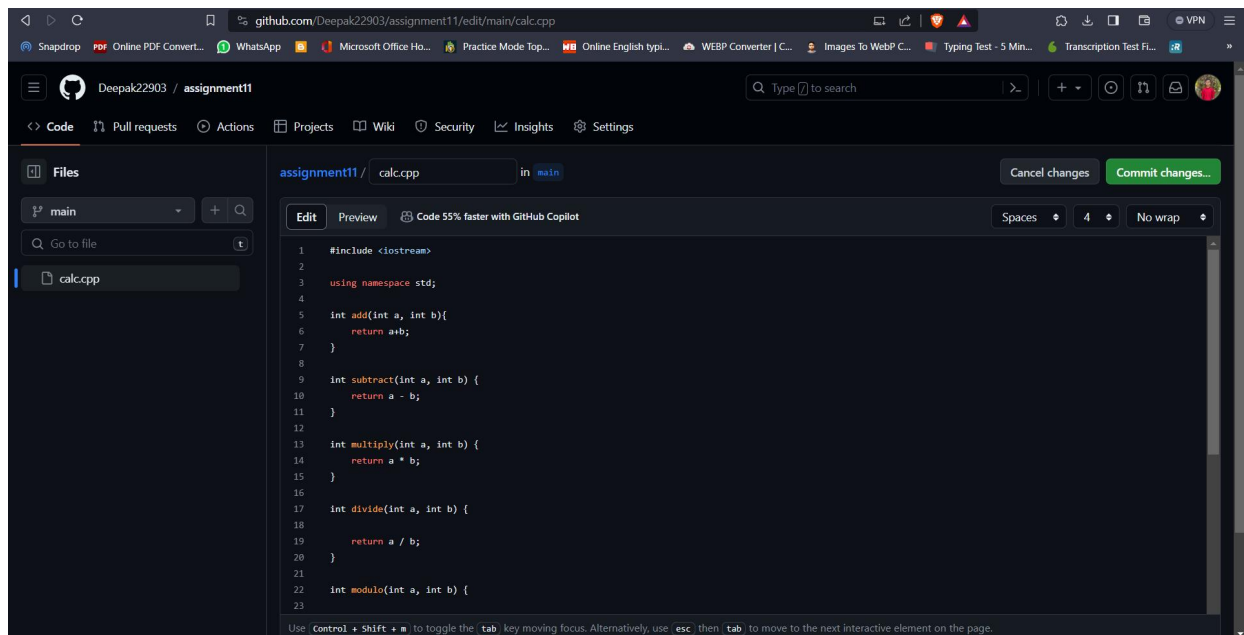
➤ Creating Repo



➤ Issue Raised



➤ Editing Forked Repo



➤ Committing Changes

Commit changes

Commit message

Added Addition Function calc.cpp

Extended description

Add an optional extended description..

☒ Commit directly to the `main` branch

☐ Create a **new branch** for this commit and start a pull request
[Learn more about pull requests](#)

Cancel

Commit changes