



Name of the Department: CSE-CCS2

Semester/Year: 5<sup>th</sup> semester

Name of the Student: SAMRUDH D YASH

Roll no: 20201CCS0070

Section: 5-CCS-2

Date & Time: 22-11-2022 :2 to 4 PM

Course code: CSE3078

Course Title: Cryptography and network security

## Book /Article Review(Self- Learning)

Book review as a teaching-learning method on course-related topics was used to gain in-depth knowledge of the relevant field with the help of books/e-books/online resources. Book/Article review helps the teacher to orient slow and advanced learners towards research and current news by incorporating scientific review. This teaching pedagogy helped the students to understand the importance of scientific learning through a book review. This set of learning skills provided the scientific update which is needed for research at the post-graduate level and helped them to understand the importance of the literature review. The following is the list of students who have used book/article review as one of the tools for the teaching-learning method.

## *Network Security Applications.*

Introduction:

### Applications of cryptography in network security

Current cryptographic algorithms are hard to understand, some of them use complex mathematics as their basic building blocks. Sometimes, this fact makes it difficult for us to see how we can apply these algorithms to provide security in a network.

So, let's start by defining what cryptography is.

“Cryptography encompasses the study of methods and techniques to secure information, so people without required permissions cannot have access to it.” Source.

Notice that what we want to achieve is to secure information. This is usually done through an information security system. This type of system usually involves certain components, such as:

- Internet
- Intrusion Detection System
- Demilitarized zone
- Firewall

  
REGISTRAR  


[Type here]

[Type here]

[Type here]



- Local Area Net
- Firewall, backup, internal servers, etc.

Now, let me explain how some of the components are implemented by using cryptographic techniques.

## Applications of Cryptography in Authentication

We need to have user authentication mechanisms in place to determine who has access to the network. These mechanisms are used at different levels of services.

For instance, a firewall won't allow a client to connect to a network or service if the user is not authenticated, or if the authenticated user does not have permission to access the specific service.

We can apply cryptography in different ways to provide user authentication mechanisms.

User authentication can be divided into two steps.

The first is to present credentials. This is when the user presents a card, ID document, user and password, etc.

The second step is verification. This step is when the credentials are verified, to determine if they are not fake and the level of access the user has.

## Workstation Authentication

Probably the most used method is the user and password combination. This one is used mostly in operating systems to determine if you have access or not.

Operating systems, like Windows or Mac OS, do not store the passwords in plain text. The reason behind this is that anyone with access to the system (sometimes computers are shared in the workplace and at home) will have access to the password of other users, which is a big security flaw.

Instead, they usually create a cryptographic hash of the password. What the operating system store is the hash instead of the plain text.

When a user presents the credentials, the operating system creates the hash of the password the user entered and compares it with the hash stored in the system. If they match, the user gets access to the system.

## Distributed environment authentication

Another situation that is related to user authentication is when there are several computers in a distributed environment, there are also several services, and client computers need access to the services.

In this scenario, the servers distributed throughout the network cannot trust that the clients can properly identify the users. The reason behind it is because of the following threats:

[Type here]

[Type here]



[Type here]



Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

- “A user may gain access to a particular workstation and pretend to be another user operating from that workstation.
- A user may alter the network address of a workstation so that the requests sent from the altered workstation appear to come from the impersonated workstation.
- A user may eavesdrop on exchanges and use a replay attack to gain entrance to a server or to disrupt operations.” Cryptography and Network Security by W. Stalling.

One of the solutions to this situation is Kerberos: the network authentication protocol. Computer scientists from MIT created Kerberos in the 1980s.

Kerberos uses symmetric cryptography and a Key Distribution Center (KDC) to authenticate users in a distributed network environment.

## Personal identity verification

Another type of authentication is based on (smart) cards that a user can possess.

There is a standard for Personal Identity Verification of Federal Employees and Contractors created by NIST. The purpose of this document is to establish “a standard for a Personal Identity Verification (PIV) system that meets the control and security objectives of Homeland Security Presidential Directive-12. It is based on secure and reliable forms of identity credentials issued by the Federal Government to its employees and contractors.”

The standard has cryptographic specifications for:

- PIV authentication key. A mandatory asymmetric private key that supports card and cardholder authentication for an interoperable environment.
- Asymmetric card authentication key. A mandatory private key that supports card authentication for an interoperable environment.
- Symmetric card authentication key (deprecated)\_An\_optional symmetric key that supports physical access.
- Digital signature key. An asymmetric private key that supports document signing.
- Key management key. An asymmetric private key that supports key establishment.
- PIV Card application administration key. An optional symmetric key used for personalization and post-issuance activities.
- PIV secure messaging key. An optional asymmetric private key that supports key establishment for secure messaging and card authentication for physical access.

The PIV Card store private keys and corresponding public key certificates and perform cryptographic operations using the asymmetric private keys.

[Type here]

[Type here]

  
REGISTRAR  


[Type here]



## Applications of Cryptography in Web Security

There are many cryptographic applications in web security. Here, I will just mention a few without going into much detail. But you can follow the links provided to go deeper into each topic.

### SSL/TLS

Authenticated Encryption (AE) is an encryption approach that addresses confidentiality and authenticity at the same time. The approach is to generate two keys, calculate the message authentication code using the first key, and encrypt the message plus the message authentication code using the second key.

SSL/TLS protocol uses this approach.

Some of the algorithms defined in the specification for the SSH transport layer protocol are:

- 3des-cbc, REQUIRED, three-key 3DES in CBC mode
- blowfish-cbc, OPTIONAL, Blowfish in CBC mode
- twofish256-cbc, OPTIONAL, Twofish in CBC mode, with a 256-bit key
- twofish-cbc, OPTIONAL, alias for “twofish256-cbc”
- twofish192-cbc, OPTIONAL, Twofish with a 192-bit key
- twofish128-cbc, OPTIONAL, Twofish with a 128-bit key
- aes256-cbc, OPTIONAL, AES in CBC mode, with a 256-bit key
- aes192-cbc, OPTIONAL, AES with a 192-bit key
- aes128-cbc, RECOMMENDED, AES with a 128-bit key

### HTTPS

“Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). In HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS) or, formerly, Secure Sockets Layer (SSL).” Source.

So, HTTPS also uses the cryptographic protocols embedded into SSL/TLS.

### SSH

One of the protocols we can use to securely communicate in an unsecured network is Secure Shell (SSH).

SSH is the protocol that most people choose to use for remote login and tunneling.

This protocol uses asymmetric cryptography to encrypt the communication between the client and the server.

Specifically, it uses the Diffie-Hellman Key exchange to securely exchange the cryptographic keys.





# PRESIDENCY UNIVERSITY

Presidency University Act, 2013 of the Karnataka Act No. 41 of 2013 | Established under Section 2(f) of UGC Act, 1956  
Approved by AICTE, New Delhi



Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

Geo tagged Data image:



Signature of the Faculty

[Type here]

[Type here]



[Type here]

Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064



# PRESIDENCY UNIVERSITY

Presidency University Act, 2013 of the Karnataka Act No. 41 of 2013 | Established under Section 2(f) of UGC Act, 1956  
Approved by AICTE, New Delhi



Name of the Department: CSE-CCS1

Semester/Year: 5<sup>th</sup> semester

Name of the Student: NIMMALA PAVAN KUMAR

Roll no: 20201CCS0012

Section: 5-CCS-1

Date & Time: 28-11-2023. 2PM to 4 PM

Course code: CSE3078  
security

Course Title: Cryptography and network

### Book /Article Review(Self- Learning)

Book review as a teaching-learning method on course-related topics was used to gain in-depth knowledge of the relevant field with the help of books/e-books/online resources. Book/Article review helps the teacher to orient slow and advanced learners towards research and current news by incorporating scientific review. This teaching pedagogy helped the students to understand the importance of scientific learning through a book review. This set of learning skills provided the scientific update which is needed for research at the post-graduate level and helped them to understand the importance of the literature review. The following is the list of students who have used book/article review as one of the tools for the teaching-learning method.

### *Man –in the Middle Attack*

Introduction:

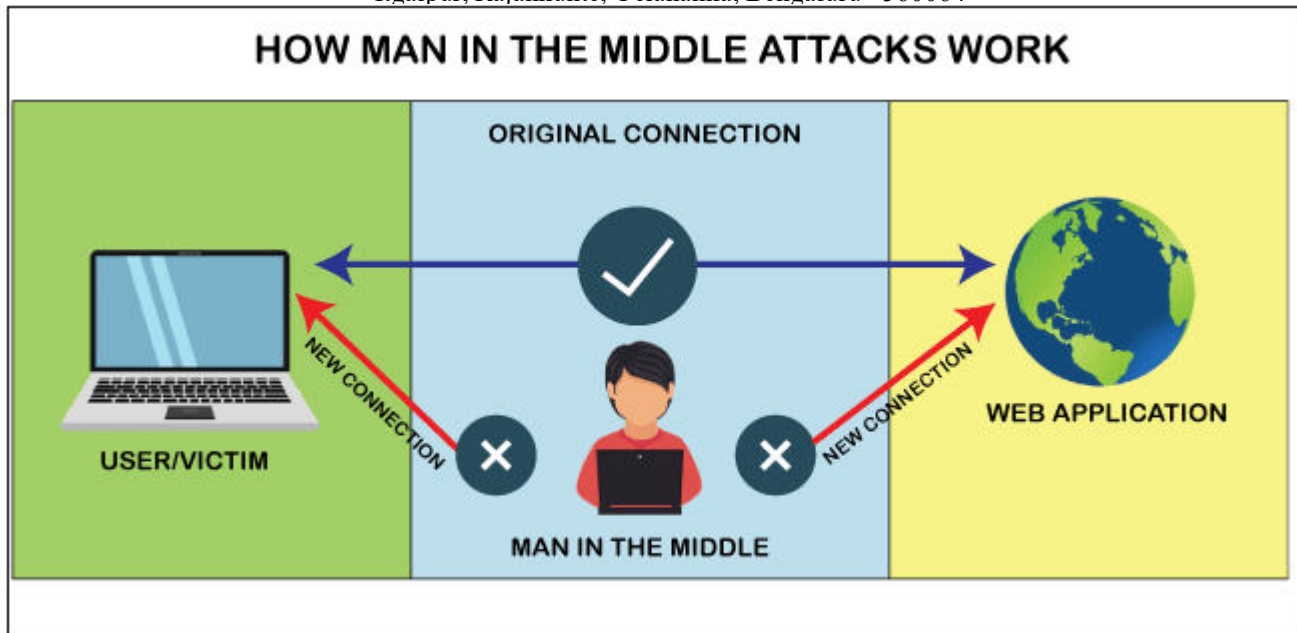
  
REGISTRAR  


[Type here]

[Type here]

[Type here]





What is MITM attack

A man in the middle (MITM) attack is a general term for when a perpetrator positions himself in a conversation between a user and an application—either to eavesdrop or to impersonate one of the parties, making it appear as if a normal exchange of information is underway.

The goal of an attack is to steal personal information, such as login credentials, account details and credit card numbers. Targets are typically the users of financial applications, SaaS businesses, e-commerce sites and other websites where logging in is required.

Information obtained during an attack could be used for many purposes, including identity theft, unapproved fund transfers or an illicit password change.

Additionally, it can be used to gain a foothold inside a secured perimeter during the infiltration stage of an advanced persistent threat (APT) assault.

Broadly speaking, a MITM attack is the equivalent of a mailman opening your bank statement, writing down your account details and then resealing the envelope and delivering it to your door.

[Type here]

[Type here]



[Type here]

## Man in the middle attack example

MITM attack progression

Successful MITM execution has two distinct phases: interception and decryption.

### Interception

The first step intercepts user traffic through the attacker's network before it reaches its intended destination.

The most common (and simplest) way of doing this is a passive attack in which an attacker makes free, malicious WiFi hotspots available to the public. Typically named in a way that corresponds to their location, they aren't password protected. Once a victim connects to such a hotspot, the attacker gains full visibility to any online data exchange.

Attackers wishing to take a more active approach to interception may launch one of the following attacks:

- **IP spoofing** involves an attacker disguising himself as an application by altering packet headers in an IP address. As a result, users attempting to access a URL connected to the application are sent to the attacker's website.
- **ARP spoofing** is the process of linking an attacker's MAC address with the IP address of a legitimate user on a local area network using fake ARP messages. As a result, data sent by the user to the host IP address is instead transmitted to the attacker.
- **DNS spoofing**, also known as DNS cache poisoning, involves infiltrating a DNS server and altering a website's address record. As a result, users attempting to access the site are sent by the altered DNS record to the attacker's site.

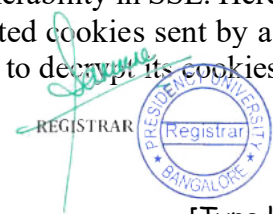
### Decryption

After interception, any two-way SSL traffic needs to be decrypted without alerting the user or application. A number of methods exist to achieve this:

- **HTTPS spoofing** sends a phony certificate to the victim's browser once the initial connection request to a secure site is made. It holds a digital thumbprint associated with the compromised application, which the browser verifies according to an existing list of trusted sites. The attacker is then able to access any data entered by the victim before it's passed to the application.
- **SSL BEAST** (browser exploit against SSL/TLS) targets a TLS version 1.0 vulnerability in SSL. Here, the victim's computer is infected with malicious JavaScript that intercepts encrypted cookies sent by a web application. Then the app's cipher block chaining (CBC) is compromised so as to decrypt its cookies and authentication tokens.

[Type here]

[Type here]



[Type here]





- **SSL hijacking** occurs when an attacker passes forged authentication keys to both the user and application during a TCP handshake. This sets up what appears to be a secure connection when, in fact, the man in the middle controls the entire session.
- **SSL stripping** downgrades a HTTPS connection to HTTP by intercepting the TLS authentication sent from the application to the user. The attacker sends an unencrypted version of the application's site to the user while maintaining the secured session with the application. Meanwhile, the user's entire session is visible to the attacker.
- **Geo tagged Data image:**



Signature of the Faculty  
Dr.A.VIJAYAKUMAR

Course: Artificial Intelligence & neural network.  
Course code: 3006

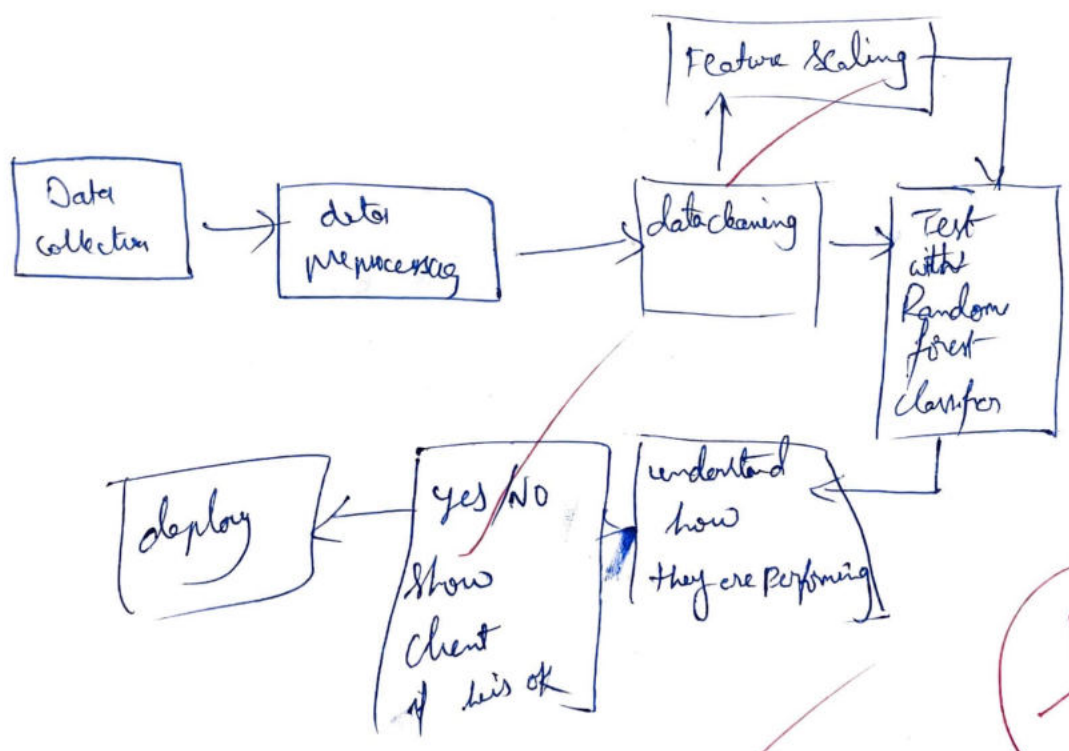
AINN

Rollno: 20191C620019

Movie Ticket Pricing System  
Develop this application using suitable data collection procedure & appropriate ML model

Developing a movie ticket pricing system involves collecting data on various factors that influence the price of a movie ticket such as time of day, day of week, movie genre, popularity of the movie, and location of the theater. This data can be collected using online databases, websites etc.

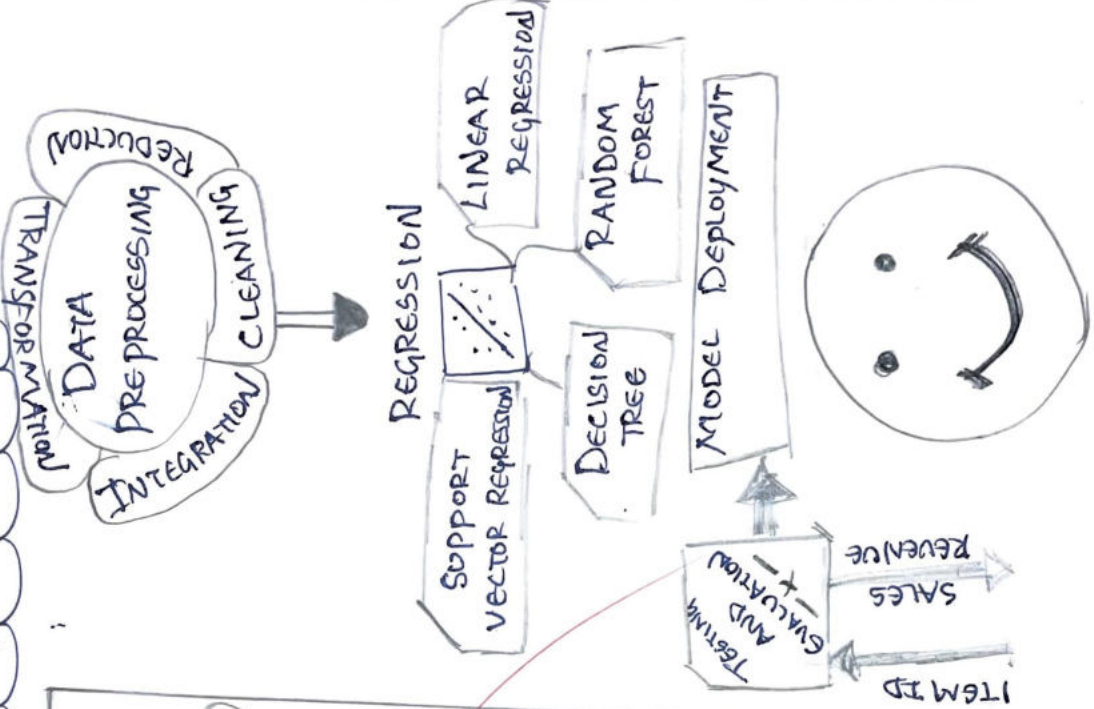
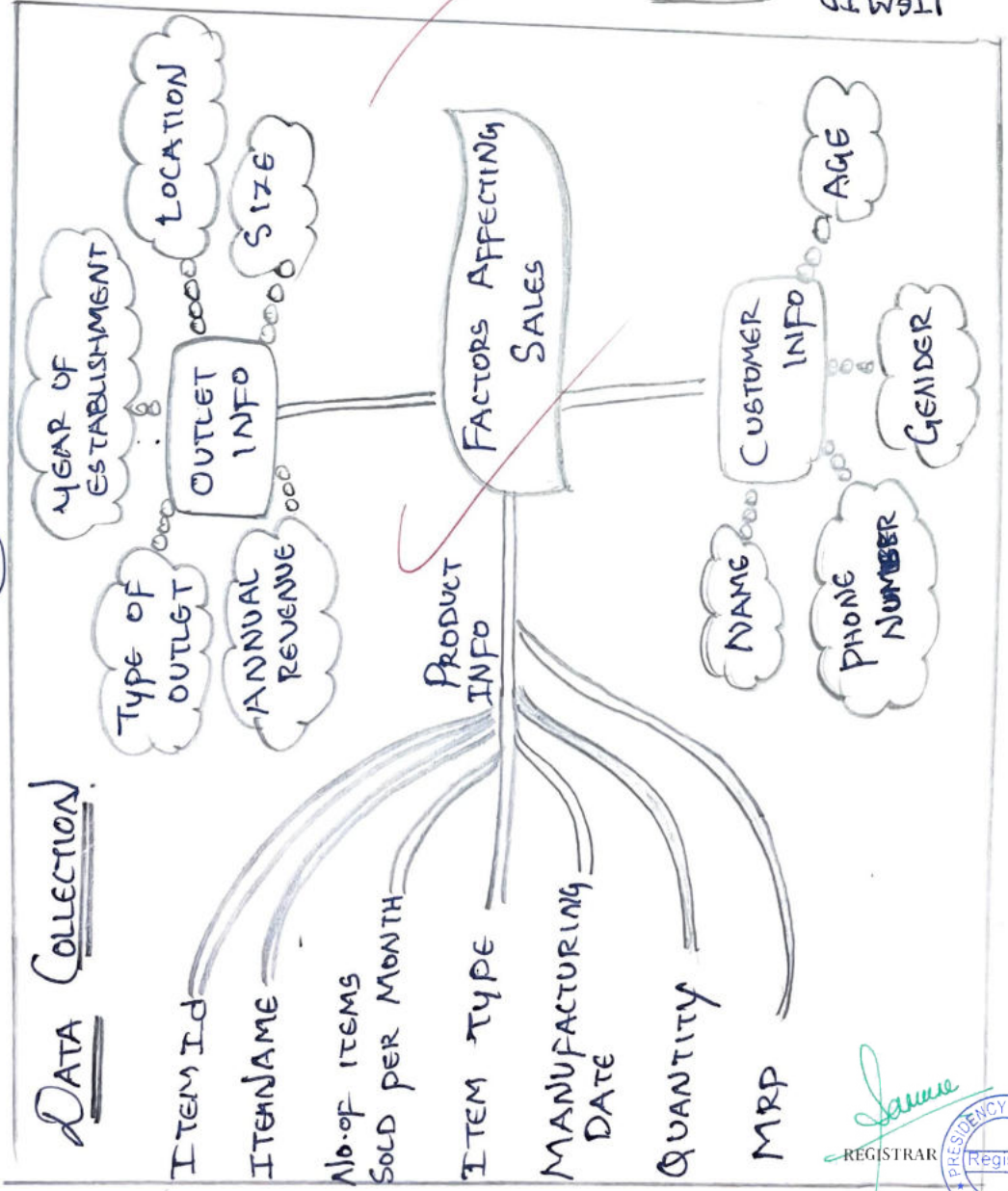
Once the data is collected, we use Random Forest Regression model, which is a type of ensemble learning model algorithm that combines decision trees to make prediction.



10 / 15

*Asha*  
REGISTRAR  
PRESIDENCY UNIVERSITY  
BANGALORE

# BIG MARKET SALES PREDICTION



REGISTRAR  
 PRESIDENCY UNIVERSITY  
 BANGALORE





Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

Name of the Department: CSE

Semester/Year: 5<sup>th</sup> semester

Name of the Student: Sudheer Kumar Reddy

Roll no: 20191CSE0894

Section: 12

Date & Time: 29-9-2021

Course code: CSE305

Course Title: Parallel Computing

## Book /Article Review (Self- Learning)

Book review as a teaching-learning method on course-related topics was used to gain in-depth knowledge of the relevant field with the help of books/e-books/online resources. Book/Article review helps the teacher to orient slow and advanced learners towards research and current news by incorporating scientific review. This teaching pedagogy helped the students to understand the importance of scientific learning through a book review. This set of learning skills provided the scientific update which is needed for research at the post-graduate level and helped them to understand the importance of the literature review. The following is the list of students who have used book/article review as one of the tools for the teaching-learning method.

## ***Recent advancement in Message Passing Interface***

MPI is a widely used communication protocol and programming model for parallel computing. It is primarily used in high-performance computing (HPC) to enable communication and coordination among multiple processes running on different nodes of a distributed computing system.

Message passing is a paradigm used widely on certain classes of parallel machines, especially those with distributed memory. Although there are many variations, the basic concept of processes communicating through messages is well understood. Over the last ten years, substantial progress has been made in casting significant applications in this paradigm. Each vendor has implemented its own variant. More recently, several systems have demonstrated that a message passing system can be efficiently and portably implemented. It is thus an appropriate time to try to define both the syntax and semantics of a core of library routines that will be useful to a wide range of users and efficiently implementable on a wide range of computers.

Benefits of the message passing interface

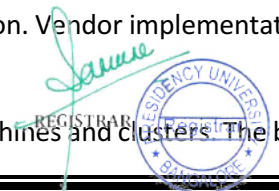
The message passing interface provides the following benefits:

- Standardization. MPI has replaced other message passing libraries, becoming a generally accepted industry standard.
- Developed by a broad committee. Although MPI may not be an official standard, it's still a general standard created by a committee of vendors, implementors and users.
- Portability. MPI has been implemented for many distributed memory architectures, meaning users don't need to modify source code when porting applications over to different platforms that are supported by the MPI standard.
- Speed. Implementation is typically optimized for the hardware the MPI runs on. Vendor implementations may also be optimized for native hardware features.
- Functionality. MPI is designed for high performance on massively parallel machines and clusters. The basic MPI-1 implementation has more than 100 defined routines.

[Type here]

[Type here]

[Type here]



Some organizations are also able to offload MPI to make their programming models and libraries faster.

Some of the key features and advancements in MPI up to September 2021 include:

1. MPI-3: The MPI-3 specification was released in 2012, which introduced several new features, including improved support for parallel I/O, enhanced one-sided communication, nonblocking collective operations, and improved support for dynamic process management.
2. MPI implementations: Various implementations of MPI exist, such as Open MPI, MPICH, and Intel MPI, among others. These implementations continually undergo improvements and optimizations for performance and scalability on different hardware architectures.
3. Performance optimizations: Researchers and developers continue to work on optimizing the performance of MPI libraries and communication patterns. These optimizations aim to reduce communication overhead, exploit network topology, and enhance scalability to support large-scale parallel applications.
4. Fault tolerance: Efforts have been made to enhance fault tolerance capabilities in MPI. This involves mechanisms to handle node failures, network partitions, and process recovery, allowing applications to continue execution despite failures.
5. Hybrid models: MPI is often used in conjunction with other programming models, such as OpenMP for shared-memory parallelism or CUDA for GPU computing. Researchers are exploring ways to improve the interoperability and performance of hybrid programming models.
6. Exascale computing: With the ongoing development of exascale supercomputers (capable of performing at least one exaFLOP, or a billion billion calculations per second), MPI is being adapted and optimized to meet the challenges posed by these massive computing systems. This includes addressing issues related to power consumption, resilience, and data movement.

It's important to note that the field of HPC and MPI is constantly evolving, and new advancements may have been made since my last knowledge update. To stay up-to-date with the latest developments, I recommend referring to research papers, technical conferences, and MPI-related websites and forums.

[Type here]

[Type here]

  
REGISTRAR  


[Type here]



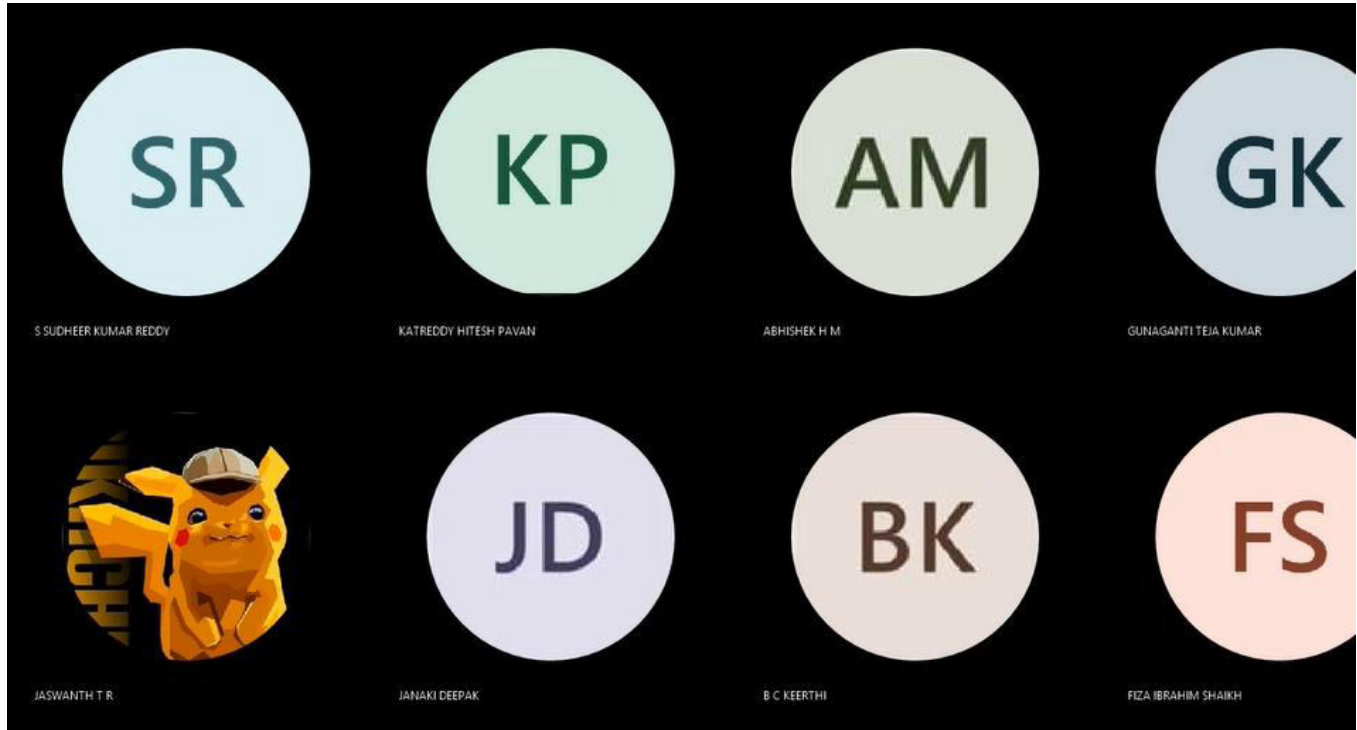
# PRESIDENCY UNIVERSITY

Presidency University Act, 2013 of the Karnataka Act No. 41 of 2013 | Established under Section 2(f) of UGC Act, 1956  
Approved by AICTE, New Delhi



Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

Online Activity Image:



Signature of the Faculty



[Type here]

[Type here]

[Type here]





# PRESIDENCY UNIVERSITY

(Established under the Presidency University Act, 2013 of the Karnataka Act 41 of 2013)

## SCHOOL OF ENGINEERING

CSE399-PROGRAMMING IN GO

### Assignment-1(Module2)

#### **Instructions:**

- Each Module assignments are part of Continuous assessment (10 marks).
- Completion of assignment is mandatory to attend Midterm and Final exams.
- The assignment submission format is attached with questions, Strictly Follow the same format.
- You can upload assignment only in MS teams.
- Late submissions will invite mark deduction
- Screen shot of customized output is mandatory.

Name: Karthik Banjan

Section: 7CSE5

Batch: B1

Instructor: Asst. Prof. Jobin S Thomas

GitHub Link: <https://github.com/karthikbanjan/399-Go-Course/tree/master/Assignment%201>



1. Ask the user to input a sentence which contains at-least 4 words. Capitalize first letter of each word and display complete sentence.

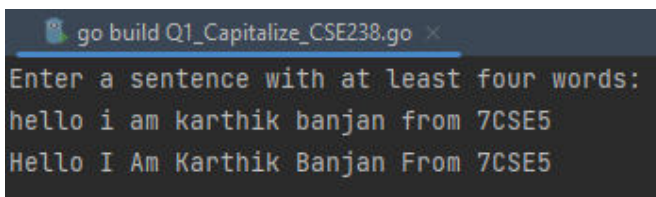
Code (typing):

```
package main

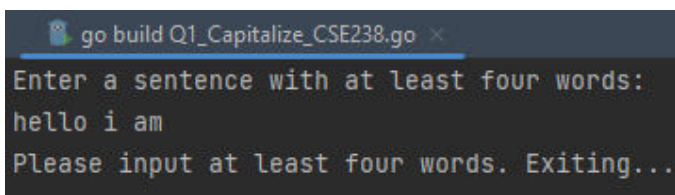
import (
    "bufio"
    "fmt"
    "os"
    "strings"
)

func main() {
    fmt.Println("Enter a sentence with at least four words:")
    inputReader := bufio.NewReader(os.Stdin)
    input, _ := inputReader.ReadString('\n')
    words := strings.Fields(input)
    if len(words) < 4 {
        fmt.Println("Please input at least four words. Exiting...")
        return
    }
    for i := 0; i < len(words); i++ {
        words[i] = strings.Title(words[i])
    }
    fmt.Println(strings.Join(words, " "))
}
```

Output (screen shot):



```
go build Q1_Capitalize_CSE238.go x
Enter a sentence with at least four words:
hello i am karthik banjan from 7CSE5
Hello I Am Karthik Banjan From 7CSE5
```



```
go build Q1_Capitalize_CSE238.go x
Enter a sentence with at least four words:
hello i am
Please input at least four words. Exiting...
```

2. Read two lists of floating-point numbers. Display whether

- 1) both lists are of same length or not
- 2) the elements in each list sum up to the same value
- 3) there are any values that occur in both the lists.

Code (typing):

```
package main

import "fmt"

func input(list *[]float64) {
    var num float64

    for {
        fmt.Scan(&num)

        if num == -1 {
            break
        }

        *list = append(*list, num)
    }
}

func sum(list []float64) float64 {
    var s float64

    for _, v := range list {
        s += v
    }

    return s
}

func common(list1 []float64, list2 []float64) []float64 {
    var comMap = make(map[float64]int)

    for _, v := range list1 {
        comMap[v]++
    }

    for _, v := range list2 {
        comMap[v]++
    }

    var comList []float64
```

```
for k, v := range comMap {
    if v > 1 {
        comList = append(comList, k)
    }
}

return comList
}

func main() {
    var list1 []float64
    var list2 []float64

    fmt.Println("Enter the numbers for first list(-1 to Exit):")
    input(&list1)
    fmt.Println("Enter the numbers for second list(-1 to Exit):")
    input(&list2)

    fmt.Println("\nList 1:", list1)
    fmt.Println("List 2:", list2)

    if len(list1) != len(list2) {
        fmt.Println("List length not equal!")
    } else {
        fmt.Println("List length equal.")
    }

    if sum(list1) != sum(list2) {
        fmt.Println("List sum not equal!")
    } else {
        fmt.Println("List sum equal.")
    }

    fmt.Println("Common, if any:", common(list1, list2))
}
```

Output (screen shot):

```
go build Q2_ListFloatOperations_CSE238.go x
Enter the numbers for first list(-1 to Exit):
1
1
2
3
4
-1
Enter the numbers for second list(-1 to Exit):
1
2
3
6
7
-1

List 1: [1 1 2 3 4]
List 2: [1 2 3 6 7]
List length equal.
List sum not equal!
Common, if any: [1 2 3]
```

3. Read a list of integers from the user, name it as input, until the user enters 0. Then Create 2 new lists namely

- 1). Square, where each element is a square of the corresponding element in the input list
- 2). Positive, with only positive numbers from the input list.

Code (typing):

```
package main

import "fmt"

func main() {
    var input []int
    var num int

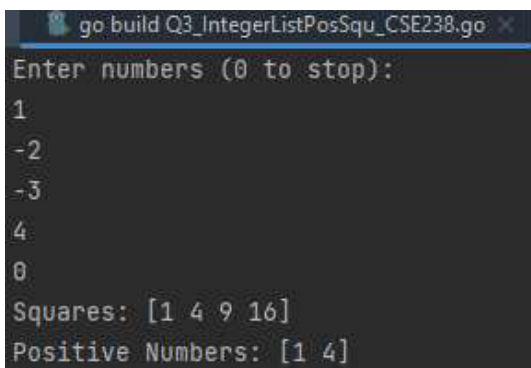
    fmt.Println("Enter numbers (0 to stop):")
    for {
        fmt.Scan(&num)
        if num == 0 {
            break
        }
        input = append(input, num)
    }

    var squareList []int
    var posList []int

    for _, v := range input {
        squareList = append(squareList, v*v)
        if v > 0 {
            posList = append(posList, v)
        }
    }

    fmt.Println("Squares:", squareList)
    fmt.Println("Positive Numbers:", posList)
}
```

Output (screen shot):



```
go build Q3_IntegerListPosSqu_CSE238.go
Enter numbers (0 to stop):
1
-2
-3
4
0
Squares: [1 4 9 16]
Positive Numbers: [1 4]
```



4. Given a map of day wise temperature in a week.

```
temp= {"sun":32, "mon":30, "tue":29,"wed":25, "thur":25, "fri":27, "sat":24}
```

Find which day recorded maximum temperature and find average temperature of the week.

Code (typing):

```
package main
```

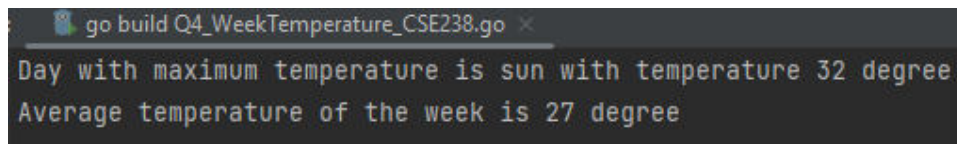
```
import (  
    "fmt"  
    "math"  
)
```

```
func main() {  
    temperature := map[string]int{"sun": 32, "mon": 30, "tue": 29, "wed": 25, "thur": 25, "fri": 27, "sat":  
24}  
    max := math.MinInt  
    var day string  
    sum := 0
```

```
    for k, v := range temperature {  
        sum += v  
        if v > max {  
            max = v  
            day = k  
        }  
    }  
}
```

```
    fmt.Println("Day with maximum temperature is", day, "with temperature", max, "degree")  
    fmt.Println("Average temperature of the week is", sum/len(temperature), "degree")  
}
```

Output (screen shot):



```
go build Q4_WeekTemperature_CSE238.go ×  
Day with maximum temperature is sun with temperature 32 degree  
Average temperature of the week is 27 degree
```

5. Initialize a map having key as section and value is another map consists of roll number and CGPA. Read roll number from the user and find section, and CGPA of the student

```
map {"cse1": map[string]float32{"cse0001":8.9, "cse003":7.8}
"cse2": map[string]float32{"cse0010":8.0, "cse0011":7.0}}
```

Code (typing):

```
package main

import "fmt"

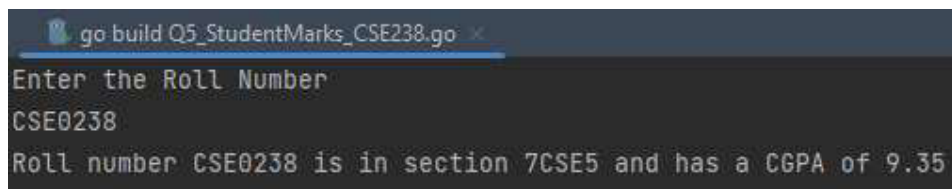
func main() {
    db := map[string]map[string]float64{"7CSE5": {"CSE0238": 9.35, "CSE0111": 8.1, "CSE0222":
8.8},
"7CSE6": {"CSE0239": 8.2, "CSE0112": 8.7, "CSE0224": 9.1}}
    var sec string
    var roll string
    var marks float64

    fmt.Println("Enter the Roll Number")
    fmt.Scan(&roll)

    for k, v := range db {
        for k2, v2 := range v {
            if k2 == roll {
                marks = v2
                sec = k
                break
            }
        }
    }

    if marks == -1 {
        fmt.Println("Roll number not found")
    } else {
        fmt.Println("Roll number", roll, "is in section", sec, "and has a CGPA of", marks)
    }
}
```

Output (screen shot):



```
go build Q5_StudentMarks_CSE238.go
Enter the Roll Number
CSE0238
Roll number CSE0238 is in section 7CSE5 and has a CGPA of 9.35
```

6. Given a map with covid patient details from a hospital with patient id and age.

data {1001:21, 1002:35, 1003:12, 1004:64, 1005:17, 1006:59, 1007:43.....}

Make 2 list, one contains **id** of young patients less than 18 years old and second one consists of **id** of senior citizens aged above 60

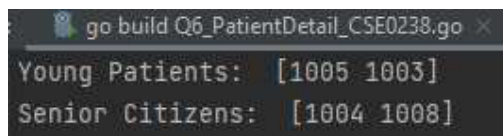
Code (typing):

```
package main
```

```
import "fmt"
```

```
func main() {  
    patientDetails := map[int]int{1001: 21, 1002: 35, 1003: 12, 1004: 64,  
        1005: 17, 1006: 59, 1007: 43, 1008: 75, 1009: 18, 1010: 60}  
    var youngPatients []int  
    var seniorPatients []int  
  
    for k, v := range patientDetails {  
        if v < 18 {  
            youngPatients = append(youngPatients, k)  
        } else if v > 60 {  
            seniorPatients = append(seniorPatients, k)  
        }  
    }  
  
    fmt.Println("Young Patients: ", youngPatients)  
    fmt.Println("Senior Citizens: ", seniorPatients)  
}
```

Output (screen shot):



```
go build Q6_PatientDetail_CSE0238.go x  
Young Patients: [1005 1003]  
Senior Citizens: [1004 1008]
```

7. Read account no., name, balance amount and amount to be deposited to create and print account details of 2 persons using struct "bank account".
8. Create account details of 2 persons using **struct "bank account"**. Read account no., name, balance amount and amount to be deposited and print the updated balance using go **functions**.

Code (typing):

```
package main

import "fmt"

type BankAccount struct {
    accountNo int
    ownerName string
    balance float64
}

func inputBankAccount() BankAccount {
    var account BankAccount
    fmt.Println("Enter account details:")
    fmt.Println("Enter account no: ")
    fmt.Scan(&account.accountNo)

    var firstName, lastName string
    fmt.Println("Enter owner's first name: ")
    fmt.Scan(&firstName)
    fmt.Println("Enter owner's last name:")
    fmt.Scan(&lastName)
    account.ownerName = firstName + " " + lastName

    fmt.Println("Enter balance: ")
    fmt.Scan(&account.balance)
    return account
}

func main() {
    acc1 := inputBankAccount()
    acc2 := inputBankAccount()

    fmt.Println(acc1)
    fmt.Println(acc2)
}
```

Output (screen shot):

```
go build Q7_BankDetailsStruct_CSE0238.go x
Enter account details:
Enter account no:
238
Enter owner's first name:
Karthik
Enter owner's last name:
Banjan
Enter balance:
5000
Enter account details:
Enter account no:
501
Enter owner's first name:
Sam
Enter owner's last name:
Banjan
Enter balance:
2000

Accounts Created:
{238 Karthik Banjan 5000} {501 Sam Banjan 2000}

Enter details of account to deposit:
Enter account no:
238
```

```
Enter details of account to deposit:
Enter account no:
238
Enter deposit amount:
50000
Account found: {238 Karthik Banjan 5000}
New balance: 55000
```

9. Write a function which takes an integer and halves it and returns true if it is even or false if it is odd. For example, half (1) should return (0, false) and half (2) should return (1, true).

Code (typing):

```
package main

import "fmt"

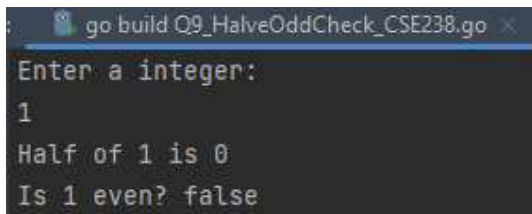
func halveCheck(num int) (int, bool) {
    return num / 2, num % 2 == 0
}

func main() {
    var num int
    fmt.Println("Enter a integer: ")
    fmt.Scan(&num)

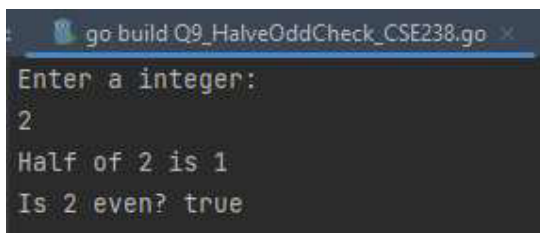
    half, isEven := halveCheck(num)

    fmt.Println("Half of", num, "is", half)
    fmt.Println("Is", num, "even?", isEven)
}
```

Output (screen shot):



```
go build Q9_HalveOddCheck_CSE238.go x
Enter a integer:
1
Half of 1 is 0
Is 1 even? false
```



```
go build Q9_HalveOddCheck_CSE238.go x
Enter a integer:
2
Half of 2 is 1
Is 2 even? true
```



10. Write a function with one variadic parameter that accepts a slice and then it finds the greatest number in a list of numbers.

Code (typing):

```
package main

import "fmt"

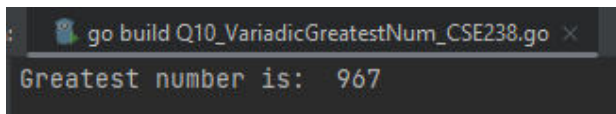
func findGreatest(num ...int) {
    max := num[0]

    for _, n := range num {
        if n > max {
            max = n
        }
    }

    fmt.Println("Greatest number is: ", max)
}

func main() {
    list := []int{545, 425, 865, 374, 583, 573, 967}
    findGreatest(list...)
}
```

Output (screen shot):



```
go build Q10_VariadicGreatestNum_CSE238.go
Greatest number is: 967
```

11. The Fibonacci sequence is defined as:  $\text{fib}(0) = 0$ ,  $\text{fib}(1) = 1$ ,  $\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2)$ . Write a recursive function in go which can find  $\text{fib}(n)$ .

Code (typing):

```
package main

import "fmt"

func fib(n int) int {
    if n == 0 {
        return 0
    }

    if n == 1 {
        return 1
    }

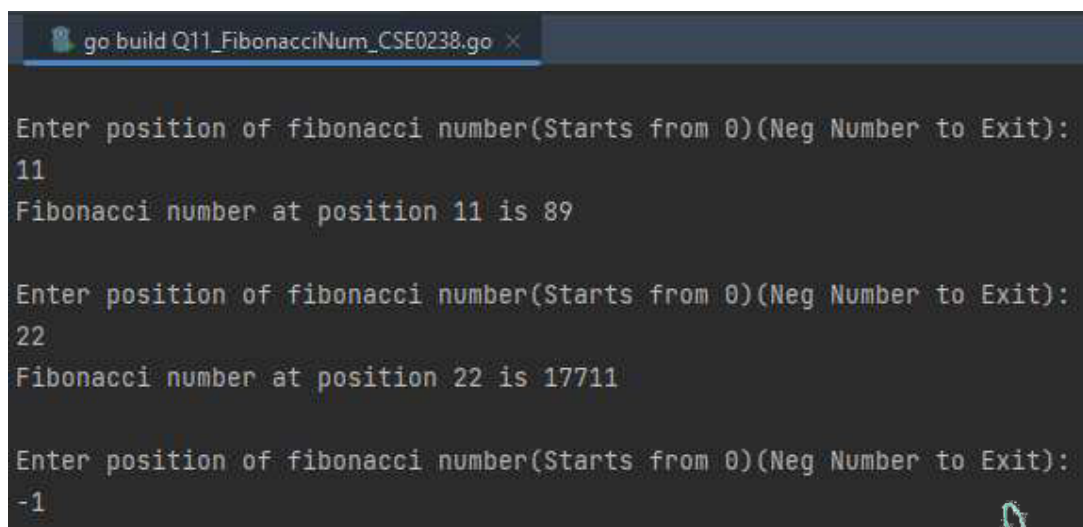
    return fib(n-1) + fib(n-2)
}

func main() {
    var n int

    for {
        fmt.Println("\nEnter position of fibonacci number(Starts from 0):")
        fmt.Scan(&n)

        fmt.Println("Fibonacci number at position", n, "is", fib(n))
    }
}
```

Output (screen shot):



```
go build Q11_FibonacciNum_CSE0238.go x
Enter position of fibonacci number(Starts from 0)(Neg Number to Exit):
11
Fibonacci number at position 11 is 89

Enter position of fibonacci number(Starts from 0)(Neg Number to Exit):
22
Fibonacci number at position 22 is 17711

Enter position of fibonacci number(Starts from 0)(Neg Number to Exit):
-1
```

Name: Anusha M.P

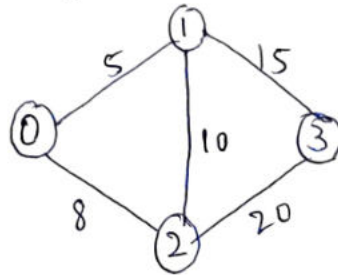
Roll No: 20211CSE0640

Section: 4CSE-13

## Design and analysis of algorithms

- CSE2007

1] Find minimum cost spanning tree of the given graph using kruskal's algorithm

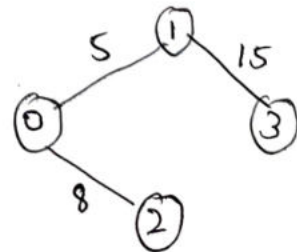


Arrange in increasing order

(0,1)	(0,2)	(1,2)	(1,3)	(2,3)
5	8	10	15	20

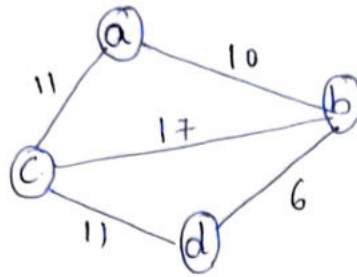
(0,1)	(0,2)	(1,2)	(1,3)	(2,3)
(0,2)	(1,2)	(1,3)	(2,3)	
(1,3)	(1,2)	(2,3)		
	x	x		

minimum cost =  $5 + 8 + 15 = 28$

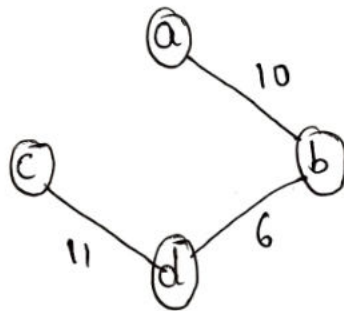


08  
10

2] Find minimum cost spanning tree of the given graph using prim's algorithm



$a(-, -)$        $b(a, 10)$        $c(a, 11)$        $d(-, \infty)$   
 $b(a, 10)$        $c(a, 11)$        $d(b, 6)$   
 $d(b, 6)$        $c(d, 11)$   
 $c(d, 11)$



Minimum Cost =  $10 + 6 + 11 = 27$

3] Apply dynamic programming techniques to solve Knapsack problem with weights  $w_1, w_2, w_3 = 1, 2, 3$  and profits  $P_1, P_2, P_3 = 10, 15, 40$  with the capacity

item	weight	Profit
1	1	10
2	2	15
3	3	40

$$N=3, M=4, w_1=1, w_2=2, w_3=3$$

$$P_1=10, P_2=15, P_3=40$$

Step 1: Start with initial condition  $i=0$  and  $j=0$

$$v[i,j]=0$$

	0	1	2	3	4
0	0	0	0	0	0
1	0	10	10	10	10
2	0	10	15	25	25
3	0	10	15	40	50

Step 2: Select the first object  $i=1, w_1=1, P_1=10$  compute the result for various value of  $j=1,2,3,4$

$$i=1$$

$w_i$	$j$	$v[i,j]$
1	1	$v[1,1] = \max(v[0,1], v[0,0]+10) = \max(0, 0+10) = 10$
1	2	$v[1,2] = \max(v[0,2], v[0,1]+10) = \max(0, 0+10) = 10$
1	3	$v[1,3] = \max(v[0,3], v[0,2]+10) = \max(0, 0+10) = 10$
1	4	$v[1,4] = \max(v[0,4], v[0,3]+10) = \max(0, 0+10) = 10$

$$i=2$$

$w_i$	$j$	$v[i,j]$
2	1	$v[2,1] = v[1,1] = 10$
2	2	$v[2,2] = \max(v[1,2], v[1,0]+15) = \max(10, 0+15) = 15$
2	3	$v[2,3] = \max(v[1,3], v[1,1]+15) = \max(10, 10+15) = 25$
2	4	$v[2,4] = \max(v[1,4], v[1,2]+15) = \max(10, 10+15) = 25$



$$\underline{i=3}$$

$$w_i \quad j \quad v[i, j]$$

$$3 \quad 1 \quad v[3, 1] = v[2, 1] = 10$$

$$3 \quad 2 \quad v[3, 2] = v[2, 2] = 15$$

$$3 \quad 3 \quad v[3, 3] = \max(v[2, 3], v[2, 0] + 40) = \max(25, 0 + 40) = 40$$

$$3 \quad 4 \quad v[3, 4] = \max(v[2, 4], v[2, 1] + 40) = \max(25, 10 + 40) = 50$$

50  $\rightarrow$  optimal solution

$$v[i, j] = v[n, m]$$

$$v[3, 4] = 50$$

To find which objected is selected

$$v[3, 4] \neq v[2, 4] \quad [\text{selected}]$$

$$50 \neq 25$$

$$x[3] = 1$$

$$j = j - w[i] = 4 - w[3] = 4 - 3 = 1$$

$$i = i - 1 = 3 - 1 = 2$$

$$v[2, 1] = v[1, 1] \quad [\text{Not Selected}]$$

$$10 = 10$$

$$j = 1$$

$$i = i - 1 = 2 - 1 = 1$$

$$v[1, 1] \neq v[0, 1] \quad [\text{Selected}]$$

$$10 \neq 0$$



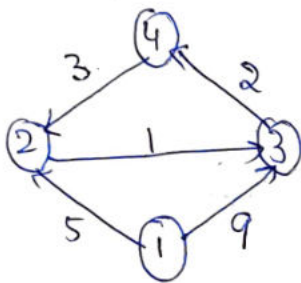
$$x[1] = 1$$

$$j = j - w_i = 1 - w[1] = 1 - 1 = 0$$

$$i = i - 1 = 1 - 1 = 0$$

$$\begin{matrix} x_1 & x_2 & x_3 \\ 1 & 0 & 1 \end{matrix}$$

4) Find the all pair shortest path problem for the given graph using suitable algorithm



	1	2	3	4
1	0	5	9	$\infty$
2	$\infty$	0	1	$\infty$
3	$\infty$	$\infty$	0	2
4	$\infty$	3	$\infty$	0

Step 1: Consider 1<sup>st</sup> column and row

No edges from 1

	1	2	3	4
1	0	5	9	$\infty$
2	$\infty$	0	1	$\infty$
3	$\infty$	$\infty$	0	2
4	$\infty$	3	$\infty$	0

Step 2: Consider 2<sup>nd</sup> column and row

$$12 = 5, 23 = 1 \Rightarrow 13 = 0$$

$$D[1,3] = \min \{ D[1,3], D[1,2] + D[2,3] \} = \min \{ 9, 5 + 1 \} = 6$$

$$D[1,3] = 0$$





	a	b	c	d
a	0	4	1	3
b	$\infty$	0	$\infty$	$\infty$
c	$\infty$	2	0	1
d	$\infty$	$\infty$	$\infty$	0

Step 2: Consider 2<sup>nd</sup> column and row

No edge from b

	a	b	c	d
a	0	4	1	3
b	$\infty$	0	$\infty$	$\infty$
c	$\infty$	2	0	1
d	$\infty$	$\infty$	$\infty$	0

Step 3: consider 3<sup>rd</sup> column & row

$$ac=1, cb=2 \Rightarrow ab=3$$

$$D[a,b] = \min \{D[a,b], D[a,c] + D[c,b]\} = \min \{4, 3\}$$

$$D[a,b] = 3$$

$$ac=1, cd=1 \Rightarrow ad=2$$

$$D[a,d] = \min \{D[a,d], D[a,c] + D[c,d]\} = \min \{3, 2\}$$

$$D[a,d] = 2$$

	a	b	c	d
a	0	3	1	2
b	$\infty$	0	$\infty$	$\infty$
c	$\infty$	2	0	1
d	$\infty$	$\infty$	$\infty$	0

REGISTRAR

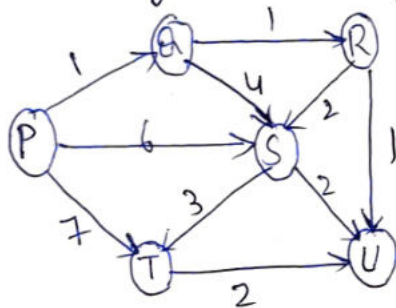


step 4: consider 4<sup>th</sup> column and row

No edges from d

	a	b	c	d
a	0	3	1	2
b	$\infty$	0	$\infty$	$\infty$
c	$\infty$	2	0	1
d	$\infty$	$\infty$	$\infty$	0

5] Find the shortest path from 'p' to all other vertices in the graph using suitable algorithm

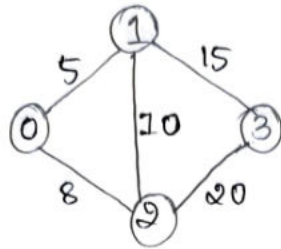


$P(-, \infty)$      $Q(P, 1)$      $R(-, \infty)$      $S(P, 6)$      $T(P, 7)$      $U(-, \infty)$   
 $Q(P, 1)$      $R(Q, 2)$      $S(P, 6)$      $T(P, 7)$      $U(-, \infty)$   
 $R(Q, 2)$      $S(R, 4)$      $T(P, 7)$      $U(R, 3)$   
 $U(R, 3)$      $S(R, 4)$      $T(P, 7)$   
 $S(R, 4)$      $T(S, 7)$   
 $T(S, 7)$

$P \rightarrow Q = 1$   
 $P \rightarrow Q \rightarrow R = 2$   
 $P \rightarrow Q \rightarrow R \rightarrow S = 4$   
 $P \rightarrow Q \rightarrow R \rightarrow S \rightarrow T = 7$   
 $P \rightarrow Q \rightarrow R \rightarrow U = 3$



1. Find Minimum cost spanning tree of the given graph using Kruskal's algorithm.



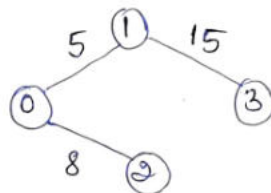
Soln

(0,1)	(0,2)	(1,2)	(1,3)	(2,3)
5	8	10	15	20
(0,1)	(0,2)	(1,2)	(1,3)	(2,3)
5	8	10	15	20
(0,2)	(1,2)	(1,3)	(2,3)	
8	10	15	20	
(1,3)	(1,2)	(2,3)		
15	10	20		

07  
10

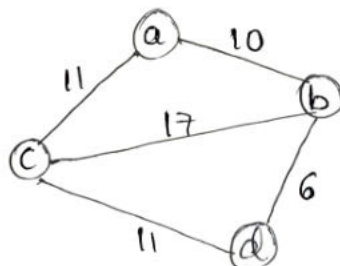
*(Signature)*

∴ The minimum spanning tree for the given graph is as follows.



cost = 28

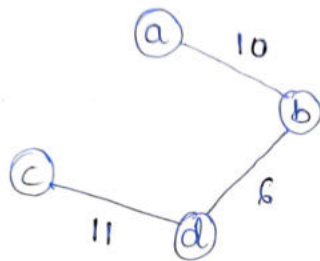
2. Find minimum cost spanning tree of the given graph using prim's algorithm.



$$V = a, b, c, d$$

ET	
a(-, -)	b(a, 10), c(a, 11), d(a, 10)
b(a, 10)	c(a, 11), d(b, 6)
d(b, 6)	c(d, 11)
c(d, 11)	

∴ The minimum spanning Tree is as follows.



The minimum spanning Tree cost is 27

3. Apply Dynamic programming techniques to solve knapsack problem with weights  $w_1, w_2, w_3 = 1, 2, 3$  and profits  $P_1, P_2, P_3 = 10, 15, 40$  with the capacity 5.

Soln

$$\begin{array}{llll}
 n=3 & w_1=1 & w_2=2 & w_3=3 \\
 m=5 & P_1=10 & P_2=15 & P_3=40
 \end{array}$$

Since we have  $n=3$  and  $m=5$  we have to maintain  $n+1$  rows i.e;  $3+1=4$  rows and  $m+1$  columns i.e;  $5+1=6$  columns

we start with the initial condition,

$$i=0 \text{ \& } j=0 \quad \therefore p(i, j) = 0.$$

$$v[i, j] = \begin{cases} 0 & \text{if } i=0, j \neq 0 \\ v[i-1, j] & \text{if } w_i > j \\ \max\{v[i-1, j], v[i-1, j-w_i]\} & \text{if } w_i \leq j \end{cases}$$





		$j \rightarrow$					
$i \downarrow$		0	1	2	3	4	5 = m
0		0	0	0	0	0	0
1		0	10	10	10	10	10
2		0	10	15	25	25	25
n=3		0	10	15	55	65	65

$\therefore 65$  is the minimum profit ✓

$i=1$

$w_i$	$j$	$v[i, j]$
1	1	$\max\{v[i-1, j], v[i-1, j-w_i] + p_i\} = \max\{v[0, 1], v[0, 0] + 10\}$ $= \max\{0, 0 + 10\} = 10$
1	2	$\max\{v[0, 2], v[0, 1] + 10\} = \max\{0, 0 + 10\} = 10$
1	3	$\max\{v[0, 3], v[0, 2] + 10\} = \max\{0, 0 + 10\} = 10$
1	4	$\max\{v[0, 4], v[0, 3] + 10\} = \max\{0, 0 + 10\} = 10$
1	5	$\max\{v[0, 5], v[0, 4] + 10\} = \max\{0, 0 + 10\} = 10$

$i=2$

$w_i$	$j$	$v[i, j]$
2	1	$\max\{v[i-1, j], v[i-1, j-w_i] + p_i\} = v[1, 1] = 10$
2	2	$\max\{v[1, 2], v[1, 0] + 15\} = \max\{10, 0 + 15\} = 15$
2	3	$\max\{v[1, 3], v[1, 2] + 15\} = \max\{10, 10 + 15\} = 25$
2	4	$\max\{v[1, 4], v[1, 3] + 15\} = \max\{10, 10 + 15\} = 25$
2	5	$\max\{v[1, 5], v[1, 4] + 15\} = \max\{10, 10 + 15\} = 25$

$i=3$

$w_i$	$j$	$v[i, j]$
3	1	$v[i-1, j] = v[2, 1] = 10$
3	2	$v[i-1, j] = v[2, 2] = 15$
3	3	$\max\{v[2, 3], v[2, 2] + 40\} = \max\{25, 15 + 40\} = 55$
3	4	$\max\{v[2, 4], v[2, 3] + 40\} = \max\{25, 25 + 40\} = 65$
3	5	$\max\{v[2, 5], v[2, 4] + 40\} = \max\{25, 25 + 40\} = 65$



$$v[3,5] \neq v[2,5]$$

$$\therefore x[3] =$$

$$j = 5 - w[3] = 5 - 3 = 2$$

$$\therefore j = 2$$

while ( $2 \neq 0$  & &  $2 \neq 0$ )

$$v[2,2] \neq v[1,2]$$

$$15 \neq 10$$

$$\therefore x[2] = 1$$

$$j = 2 - w[2] = 2 - 2 = 0$$

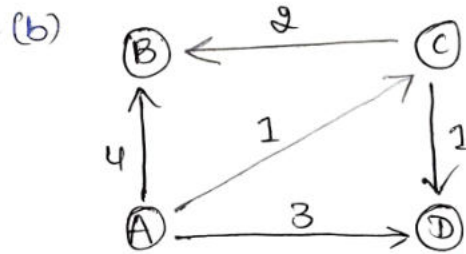
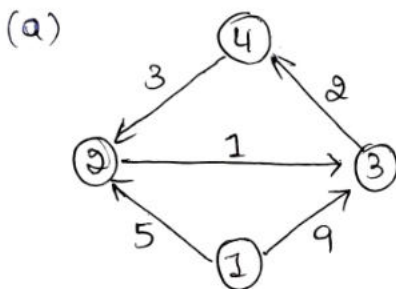
while ( $1 \neq 0$  & &  $0 \neq 0$ )  $\rightarrow$  false

$\therefore$  process ends

$$\therefore \underline{x_i = [0, 1, 1]}$$

The items 2 & 3 will be selected.

4. Find all pair shortest path problem for the given graph using suitable algorithm.



(a)

	1	2	3	4
1	0	5	9	$\infty$
2	$\infty$	0	1	$\infty$
3	$\infty$	$\infty$	0	2
4	$\infty$	3	$\infty$	0

$$1,2 = 5, \quad 2,3 = 1 \Rightarrow 1,3 = ?$$

$$1,3 = \min(1,3, 1,2 + 2,3)$$

$$= \min(9, 5 + 1)$$

$$1,3 = 6$$

$$4,2 = 3, \quad 2,3 = 1 \Rightarrow 4,3 = ?$$

$$4,3 = \min(4,3, 4,2 + 2,3)$$

$$= \min(\infty, 3 + 1)$$

$$4,3 = \underline{\underline{4}}$$

	1	2	3	4
1	0	5	6	$\infty$
2	$\infty$	0	1	$\infty$
3	$\infty$	$\infty$	0	2
4	$\infty$	3	4	0

$$(1,3)=6, (3,4)=2 \Rightarrow (1,4)=\min(\infty, 8)$$

$$(1,4)=8$$

$$(2,3)=1, (3,4)=2 \Rightarrow (2,4)=\min(\infty, 3)$$

$$(2,4)=3$$

$$(4,3)=4, (3,4)=2 \Rightarrow (4,4)=\min(0, 2)$$

$$(4,4)=0$$

	1	2	3	4
1	0	5	6	8
2	$\infty$	0	1	3
3	$\infty$	$\infty$	0	2
4	$\infty$	3	4	0

$$(1,4)=8, (4,2)=3 \Rightarrow (1,2)=\min(1, 2, 1, 4+4, 2) \\ =\min(5, 8+3)=(5, 11)$$

$$(1,2)=5$$

$$(1,4)=8, (4,3)=4 \Rightarrow (1,3)=\min(6, 12)$$

$$(1,3)=6$$

$$(2,4)=3, (4,2)=3 \Rightarrow (2,2)=\min(0, 6)$$

$$(2,2)=0$$

$$(2,4)=3, (4,3)=4 \Rightarrow (2,3)=\min(1, 7)=1$$

$$(3,4)=2, (4,2)=3 \Rightarrow (3,2)=\min(\infty, 5)$$

$$(3,2)=5$$

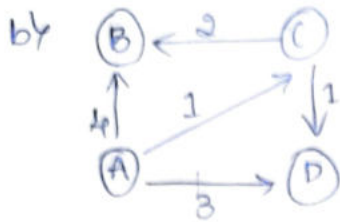
$$(3,4)=2, (4,3)=4 \Rightarrow (3,3)=\min(0, 6)$$

$$(3,3)=0$$

Hence the final matrix is

	1	2	3	4
1	0	5	6	8
2	$\infty$	0	1	3
3	$\infty$	5	0	2
4	$\infty$	3	4	0





	A	B	C	D
A	0	4	1	3
B	$\infty$	0	$\infty$	$\infty$
C	$\infty$	2	0	1
D	$\infty$	$\infty$	$\infty$	0

$$(A,C)=1 \quad (C,B)=2 \Rightarrow (A,B)=\min(4,3)$$

$$\underline{A,B=3}$$

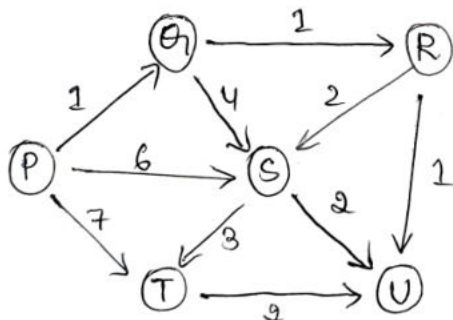
$$(A,C)=1 \quad (C,D)=1 \Rightarrow (A,D)=\min(3,2)$$

$$\underline{A,D=2}$$

Hence the final matrix is

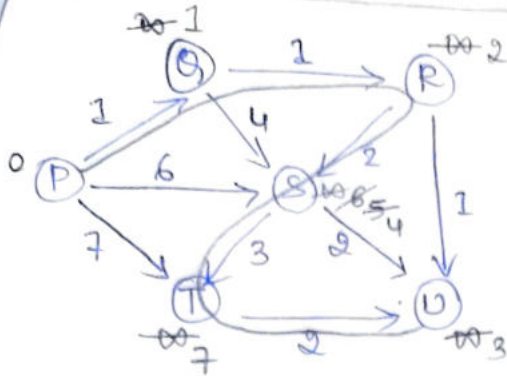
	A	B	C	D
A	0	3	1	2
B	$\infty$	0	$\infty$	$\infty$
C	$\infty$	2	0	1
D	$\infty$	$\infty$	$\infty$	0

5 Find the shortest path from 'p' to all other vertices in the graph using suitable algorithm



Soln By applying Bellman Ford Algorithm, we can solve the above graph





n vertices  
 6 vertices  
 $6-1 = 5$  times

	P	Q	R	S	T	U	
P, Q							
P, S	P	0	∞	∞	∞	∞	
P, T	Q	0	1	2	6	7	3
Q, R	R	0	1	2	5	7	3
Q, S	S	0	1	2	4	7	3
R, S	T	0	1	2	4	7	3
R, U	U	0	1	2	4	7	3
S, T							
S, U							
T, U							

∴ The shortest path is

- (costs of) → P = 0  
 Q = 1  
 R = 2  
 S = 4  
 T = 7  
 U = 3



Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

Name of the Department: CSE

Semester/Year: 3rd semester

Name of the Student: RAMANNAGARI LOKESH

Roll no: 20211CSE0071

Section: 3CSE2

Date & Time: 14-09-2022

Course code: CSE2027

Course Title: Fundamentals of Data Analytics

## Flipped Classroom

Flipping the classroom (also known as “inverting” a classroom) is a “pedagogy-first” approach to teaching in which course materials are introduced outside of class, and in-class time is re-purposed for inquiry, application, and assessment in order to better meet the needs of individual learners.

*Sampling Techniques: Fundamental Definitions, Important sampling distributions concept of standard error, Descriptive Statistics, Inferential Statistics (T test, Z test), Probability Uses In Business and Calculating Probability from a Contingency Tables.*

  
REGISTRAR  


Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064



Signature of the Faculty

[Type here]

[Type here]



[Type here]

COURSE: FDA

Code: CSE2027

① Data

8/15

\* It is the amount of information collected for a particular purpose is called Data

\* Information is the description of some known thing of particular subject.

\* Data provides the details of previous process and information of particular subject.

\* Example: Group of <sup>students</sup> people attending class → attendance of students is a Data

\* Information: Eg: It is the information of students who are attending classes

② Data Analysis Tools

a) SQL

b) JAVA

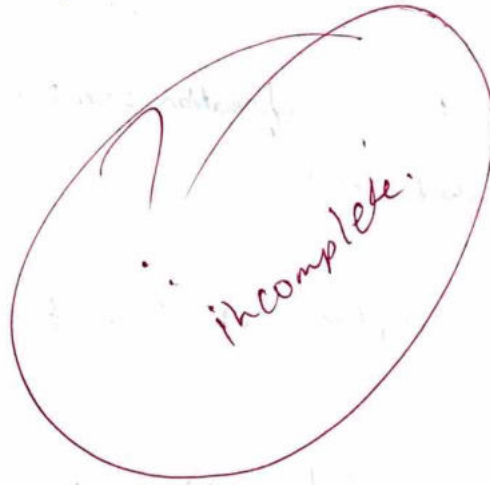
c) MATLAB

d) RDBMS database management tools

*hooi*



### ③ Types of digital data



### ④ Types of data Analysis

- + Diagnostic data Analysis
- + descriptive data Analysis
- + Prescriptive data Analysis
- + Graphical data Analysis

→ Diagnostic data Analysis - It is type of data Analysis where ~~digital~~ data will observed and decision will be made

→ descriptive data Analysis - ~~It~~ This type of data Analysis is done when the description of given data is also given Ex: Average of students scoring passing marks.

→ These V's of data defines the characteristic and structure of the data

→ Vulnerability - \* This ~~means~~ shows vulnerability a different charactel of the data

\* We can know ~~the~~ ~~how~~ how vulner ~~is~~ the data is. and whats the status of it.

→ Validity - \* This helps to know the validity of the data.

\* Helps in various data Analysis.

→ Value - \* This shows the value of the given data

\* This helps us to know whats the purpose of the data.

→ Velocity - \* the data's usage speed ~~is~~ can be known

\* Speed in the sense its range of collecting information

→ Variability - \* This shows the variability in the data

\* Helps in checking variation in data.

→ Variety - \* Shows the variety in data

\* Variety of data deals with variety of purpose.



Versatility - \* This shows ~~the~~ ~~that~~ How versatile  
does the data is.

\* Versatility ~~matter~~ of data matters in  
comparison to maintain the stability.

### ⑧ Sampling Techniques

! <sup>complex</sup>

NAME : E:DEPT 1

Roll No. : 2020ICIT0041

SUBJECT : FOG COMPUTING .

COURSE CODE : CSE 2032



(1) Explain types of Fog Computing.

→ Device-level fog computing:

It runs on devices such as sensors, switches, routers, and other low-powered hardware. It can be used to gather data from these devices and send it to the cloud for analysis.

• Edge-level fog computing:

It runs on servers or appliances located at the edge of a network. These devices can be used to process data before it is sent to the cloud.

• Gateway-level fog computing:

It runs on devices that act as a gateway between the edge and the cloud. These devices can be used to manage traffic and ensure that only relevant data is sent to the cloud.

• Cloud-level fog computing:

It runs on servers or appliances located in the cloud. These devices can be used to process data before it is sent to end users.

(2) Explain Fog Computing architecture.

⇒ Fog computing is an architectural model that extends the capabilities of cloud computing closer to the edge of the network, closer to where data is generated and consumed. It aims to address the challenges posed by the increasing number of connected devices and the growing demand for low-latency and real-time data processing.

In fog computing, computing resources and services are distributed across a hierarchical architecture, which include the cloud, the fog layer, and the edge devices. Here's a breakdown of each component:

- Cloud:

The cloud represents the centralized data centers that provide large-scale storage, processing power, and various services. It serves as the backend infrastructure for the fog computing architecture and is responsible for handling tasks that require significant computational resources and long-term storage.

- Fog layer:

The fog layer, also known as the edge or the fog node, is located closer to the edge devices and acts as an intermediate layer between the edge and the cloud. Fog nodes are typically deployed in close proximity to the edge devices, such as in local data centers, routers, or gateway devices.

- Edge Devices:

Edge devices are the endpoints of the fog computing architecture, which include sensors, actuators, smartphones, wearables, and other IoT devices. These devices generate a vast amount of data and are typically located at the network's periphery, in close proximity to the physical environment where data is collected.

10/10

Good feedback





# PRESIDENCY UNIVERSITY

Presidency University Act, 2013 of the Karnataka Act No. 41 of 2013 | Established under Section 2(f) of UGC Act, 1956  
Approved by AICTE, New Delhi



Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064



# PRESIDENCY UNIVERSITY

Presidency University Act, 2013 of the Karnataka Act No. 41 of 2013 | Established under Section 2(f) of UGC Act, 1956  
Approved by AICTE, New Delhi



Name of the Department: CSE

Semester/Year: 7<sup>th</sup> semester

Name of the Student: JOBIN THOMAS

Roll no: 20191CSE0535

Section: 7CSE10

Date & Time: 27-12-2022

Course code: CSE2033

Course Title: Go Programming

## MINI PROJECT web based server

So, In this project I have design simple web based server with the help of Go language, HTML and CSS. I try to explain what my web app is doing with the above Daigram

Below are code and screenshot of my mini Project :-.

## Index.html File

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>static website</title>
  <link rel="stylesheet" href="style.css">
  <script src="https://kit.fontawesome.com/464684a769.js"
crossorigin="anonymous"></script>
</head>
<body>
  <h2>Learn Go Programming(<span>Mr. Jobin Thomas</span></h2>
  <h3>Total Modules - 4</h3>
  <h4>Module 1 - (<span>Introduction to go programming</span></h4>
  <h4>Module 2 - (<span>Composite type and functions</span></h4>
  <h4>Module 3 - (<span>Pointer,Structs,Interface and modules</span></h4>
  <h4>Module 4 - (<span>Applications of GO</span></h4>
  <h3>REFERENCE MATERIALS</h3>
  <h4>Text Books:</h4>

```



[Type here]

[Type here]

[Type here]

Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

```
<h5>T1. Learning Go: An Idiomatic Approach to Real World Go Programming(<span>John
Badner</span>)</h5>
<h4>Reference Books:</h4>
<h5>R1. The Go Programming Language(<span>Alan A.A Donovan and Brian W.
Kernighan</span>)</h5>
<h5>R2. Mastering Go(<span>Tsoukalos M.</span>)</h5>
<H4>Web Based Resource and E-books</H4>
<h5>W1. Available at:<a href="https://www.golangprograms.com/go-language.html"><i
class="fa-solid fa-arrow-up-right-from-square"></i></a></h5>
<h5>W1. EBSCO database of Presidency ubiversity: <a
href="https://puniversity.informaticsglobal.com/login"><i class="fa-solid fa-arrow-up-
right-from-square"></i></a></h5>
<h5>W1. Go document: <a href="https://go.dev/doc/"><i class="fa-solid fa-arrow-up-
right-from-square"></i></a></h5>
<button href="/static/form.html" class="btn">Register</button>
</body>
</html>
```

## form.html File

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div>
    <form method="POST" action="/form" class="contact">
      <label>Name</label><input name="name" type="text" value=""/>
      <label>Address</label><input name="address" type="text" value=""/>

      <button class="btn1" type="submit" value="submit">Submit</button>
    </form>
  </div>
</body>
</html>
```

[Type here]

[Type here]



[Type here]

## Style.css File

```
*{
  background: rgb(0, 0, 0);
  color: #fff;
  padding: 5px;
}

span{
  font-style: italic;
}

i{
  color: #fff;
}

.btn{
  display: block;
  margin: 50px auto;
  width: fit-content;
  border: 1px solid #ff004f;
  padding: 14px 50px;
  border-radius: 6px;
  text-decoration: none;
  color: #fff;
  transition: background 0.5s;
}

.btn:hover{
  background: #ff004f;
}

.btn1{
  display: block;
  margin: 50px auto;
  width: fit-content;
  border: 1px solid #ff004f;
  border-radius: 6px;
  text-decoration: none;
  color: #fff;
  transition: background 0.5s;
}

.btn1:hover{
  background: #ff004f;
}
```





```
}  
  
.contact{  
  flex-basis: 50%;  
}  
  
form input{  
  width: 100%;  
  border: 0;  
  outline: none;  
  background: #262626;  
  padding: 15px;  
  margin: 15px 0;  
  color: #fff;  
  font-size: 18px;  
  border-radius: 6px;  
}
```

## main.go File

```
package main  
  
import (  
  "fmt"  
  "log"  
  "net/http"  
)  
  
func formHandler(w http.ResponseWriter, r *http.Request) {  
  if err:= r.ParseForm(); err !=nil {  
    fmt.Fprintf(w, "ParseForm() err: %V", err)  
    return  
  }  
  fmt.Printf(w, "Post request successful")  
  name := r.FormValue("name")  
  address := r.FormValue("address")  
  fmt.Printf(w, "Name = %S\n", name)  
  fmt.Printf(w, "Address = %S\n", address)  
}  
  
func helloHandler(w http.ResponseWriter, r *http.Request){  
  if r.URL.Path != "/hello" {  
    http.Error(w, "404 not found", http.StatusNotFound)  
    return  
  }  
}
```



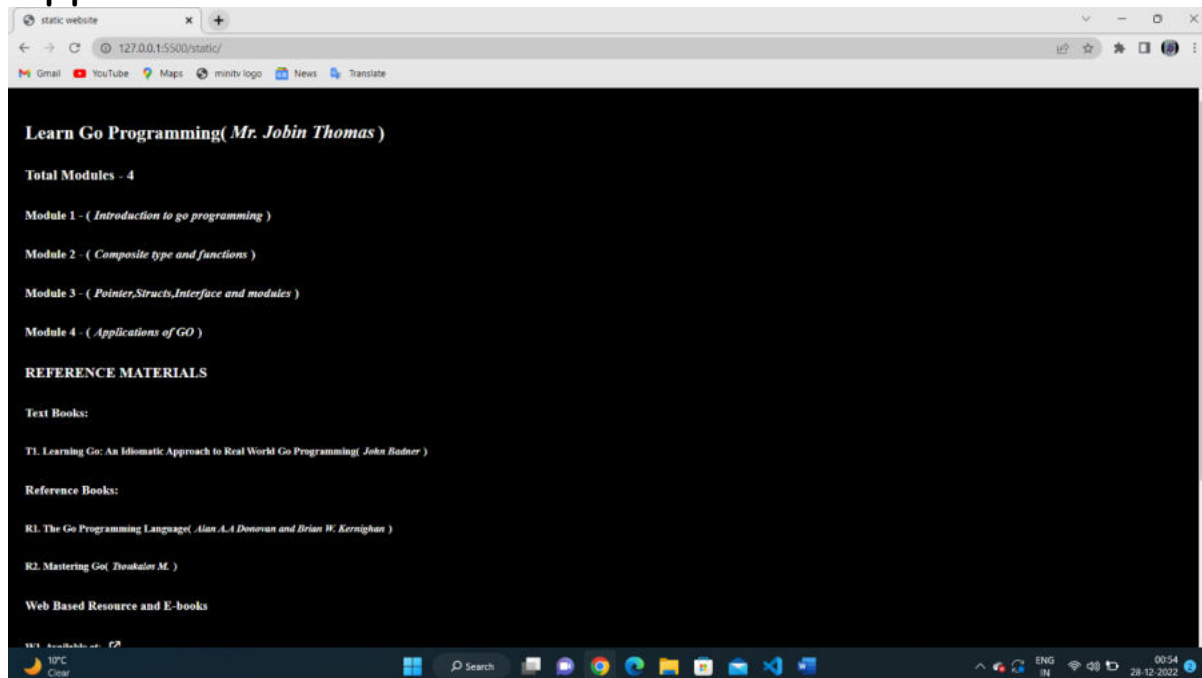
Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

```
if r.Method != "GET" {
    http.Error(w, "method is not supported", http.StatusNotFound)
    return
}
fmt.Printf(w, "hello!")
}

func main() {
    fileServer := http.FileServer(http.Dir(".static"))
    http.Handle("/", fileServer)
    http.HandleFunc("/form", formHandler)
    http.HandleFunc("/hello", helloHandler)

    fmt.Printf("Starting server at port 5500\n")
    if err := http.ListenAndServe(":5500", nil); err != nil {
        log.Fatal(err)
    }
}
```

## App Screenshots



[Type here]

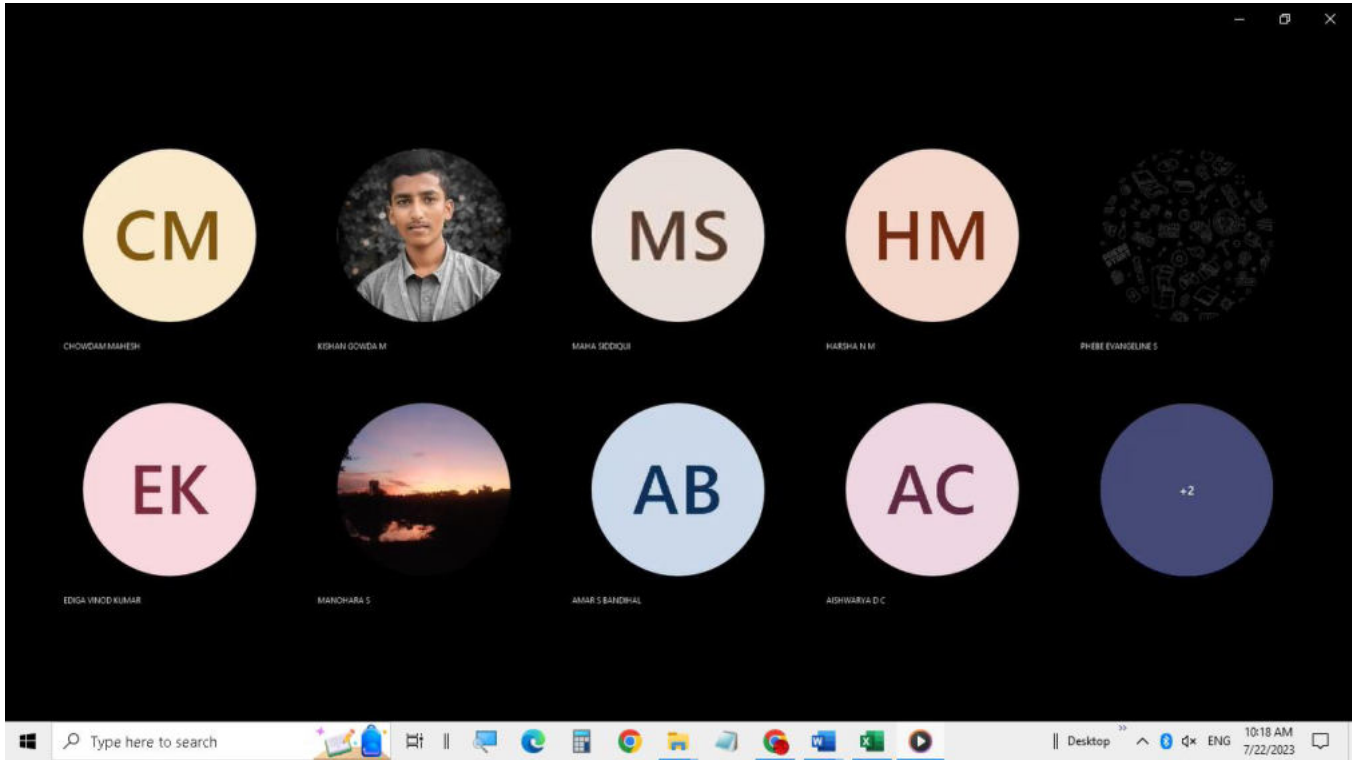
[Type here]

*Sanu*  
REGISTRAR  
PRESIDENCY UNIVERSITY  
BANGALORE

[Type here]

Itgalpur, Rajankunte, Yelahanka, Bengaluru - 560064

# Online discussion screenshot



Signature of the Faculty

[Type here]

[Type here]



[Type here]