



DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

**A
MINI PROJECT REPORT**

ON

“IMPLEMENTATION OF ANTIVIRUS USING C++”

Submitted in the partial fulfillment of the requirements in the 4th semester of

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

BY

**SOUJANYA S
1NH17IS105**

Under the guidance of

Mrs. Shwetha K S,
Sr. Assistant Professor,
Dept. of ISE, NHCE

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

NEW HORIZON COLLEGE OF ENGINEERING

(Autonomous College Permanently Affiliated to VTU, Approved by AICTE, Accredited by
NBA & NAAC with 'A' Grade)
Ring Road, Bellandur Post, Near Marathalli,
Bangalore-560103, INDIA



CERTIFICATE

Certified that, the mini project report entitled “**IMPLEMENTATION OF ANTIVIRUS USING C++**” carried out by SOUJANYA S (1NH17IS105), a bonafied student of New Horizon College of Engineering, Bengaluru, in partial fulfillment of the requirements in the IV semester of Bachelor of Engineering in Information Science and Engineering the year 2018-2019. The project report has been approved as it satisfies the academic requirement in respect of mini project work.

Name & Signature of Guide

(Mrs. Shwetha K S)

Name & signature of HOD

(Dr. R J Anandhi)

ABSTRACT

The project titled “Implementation of Antivirus using C++” is an application developed to prevent or detect malware. In today’s world, the one word that can strike fear in the heart of any computer user, especially the one who accesses the internet or exchanges diskettes is the word “VIRUS”. These viruses can generate so much fear in the cyber world that the news of a new virus often spreads faster than the virus itself. This application deals primarily with the security of the computer as well as mobile files by securing the files from virus attacks and checking a particular file for virus. It deletes all viruses from existing system and therefore makes the system more efficient and fast. This is the best way for scanning for viruses in our systems rather than purchasing external antiviruses from market. Having an antivirus application installed in our system for free is always better than paying for one.

Antivirus software scans the file comparing specific bits of code against information in its database and if it finds a pattern duplicating one in the database, it is considered a virus, and it will quarantine or delete that particular file. The above mention application uses the signature-based detection which is the most common form of detection that checks all the .EXE files and validates it with the known list of viruses and other types of malware or it checks if the unknown executable files show any misbehavior as a sign of unknown viruses. Files, programs and applications are basically scanned when they in use. Once an executable file is downloaded it is scanned for any malware instantly. Antivirus software can also be used without the background on access scanning, but it is always advisable to use on access scanning because it is complex to remove malware once it infects your system.

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project. I would like to take an opportunity to thank them all.

I thank the management, **Dr. Mohan Manghnani**, Chairman, New Horizon Educational Institutions for providing necessary infrastructure and creating good conducive environment for effective learning.

I also here record constant encouragement support and facilities extended to us by **Dr. Manjunatha**, Principal, New Horizon College of Engineering, Bengaluru.

I extend my sincere gratitude for the constant encouragement support and facilities provided to us by **Dr. R J Anandhi**, Professor and Head of the Department, Department of Information Science and Engineering, New Horizon College of Engineering, Bengaluru.

I sincerely acknowledge the encouragement, timely help and guidance to me by **Mrs. Shwetha K S, Sr. Assistant Professor**, Department of ISE, NHCE, and Bengaluru, to complete the mini project within the stipulated time.

Finally, a note of thanks to the teaching and non-teaching staff of Information Science and Engineering Department for their cooperation extended to us and our friends, who helped me directly or indirectly in the successful completion of this mini project.

SOUJANYA S
(1NH17IS105)

TABLE OF CONTENTS

Abstract	i
Acknowledgement	ii
Table of contents	iii
List of figures and tables	iv
Chapters	
Chapter 01: Introduction	01
1.1 Motivation of Project	15
1.2 Problem statement	16
Chapter 02: System Requirements and Specification	17
2.1 Hardware Specifications	18
2.2 Software specifications	18
2.3 About the Language	18
Chapter 03: Methodology	27
3.1 Flowchart	27
3.2 Algorithm	28
3.3 Code and Implementation	29
Chapter 04: Results and Discussion	32
4.1 Output snapshots	32
Conclusion and Future Enhancements	34
References	35

LIST OF FIGURES AND TABLES

Fig 1.1	AN EXECUTABLE VIRUS FILE INSTALLED IN THE SYSTEM	12
Fig 1.2	BINARY CODE OF A NORMAL FILE	13
Fig 1.3	VIRUS SIGNATURES OF A VIRUS FILE	13
Fig 1.4	VIRUS DATABASE THAT CONTAINS THE VIRUS SIGNATURES	14
Fig 4.1	WHEN THE FILE ENTERED IS NOT FOUND	32
Fig 4.2	A VIRUS FILE IS DETECTED AND DELETED	32
Fig 4.3	A VIRUS IS NOT DETECTED; THEREFORE FILE SAFE MESSAGE IS DISPLAYED.	33
Table 1.1	IDENTIFIABLE CHARACTERS IN A VIRUS FILE	14
Table 2.1	BUILT-IN DATA TYPES	20
Table 2.2	SIZES OF VARIABLES	22
Table 2.3	LOOP TYPES AND DESCRIPTION	24

Chapter 01

INTRODUCTION

The mini project “Implementation of Antivirus using C++”, is an application written in C++ to detect virus in systems and to avoid the system from singular troubles that can occur because of virus. Antivirus software can be designed to detect malicious software, which is commonly called as malware, on our systems and remove the malware software that can corrupt the system and reduce its efficiency. Since most viruses are designed to run in the background, most users do not know when their computer is infected. Antivirus programs serve to search for, detect, and remove these viruses. Therefore it is necessary, that the, Antivirus programs are kept up-to-date in order to enable them to detect new viruses and warn the user.

WHAT IS A COMPUTER VIRUS?

Robert Thomas, an engineer at BBN Technologies developed the first known computer virus in the year 1971. The first virus was christened as the “Creeper” virus, and the experimental program carried out by Thomas infected mainframes on ARPANET. The teletype message displayed on the screens read, “I’m the creeper: Catch me if you can.”

But the original wild computer virus, probably the first one to be tracked down in the history of computer viruses was “Elk Cloner.” The Elk Cloner infected Apple II operating systems through floppy disks. The message displayed on infected Apple Computers was a humorous one. The virus was developed by Richard Skrenta, a teenager in the year 1982. Even though the computer viruses were designed as a prank, it also enlightened how a malicious program could be installed in a computer’s memory and stop users from removing the program.

The term 'computer virus' was first formally defined by Fred Cohen in 1983. Computer viruses never occur naturally. They are always induced by people. Once created and released, however, their diffusion is not directly under human control.

A computer virus is a type of malicious code or program written to alter the way a computer operates and is designed to spread from one computer to another. A virus operates by inserting or attaching itself to a legitimate program or document that supports

macros in order to execute its code. In the process, a virus has the potential to cause unexpected or damaging effects, such as harming the system software by corrupting or destroying data. When this replication succeeds the computer is said to be infected or corrupted with a computer virus. The main purpose of creating a computer virus is to infect vulnerable systems and gain admin control to steal user sensitive data.

Not all computer viruses are destructive though. However, most of them perform actions that are malicious in nature, such as destroying data. Some viruses wreak havoc as soon as their code is executed, while others lie dormant until a particular event gets initiated, that causes their code to run in the computer. Viruses spread when the software or documents they get attached to are transferred from one computer to another using a network, a disk, file sharing methods, or through infected e-mail attachments. Some viruses use different stealth strategies to avoid their detection from anti-virus software. For example, some can infect files without increasing their sizes, while others try to evade detection by killing the tasks associated with the antivirus software before they can be detected. Some old viruses make sure that the last modified date of a host file stays the same when they infect the file.

HOW DOES A COMPUTER VIRUS OPERATE?

A computer virus can operate in two ways. The first one is, as soon as the virus lands on a new system it begins to replicate whereas, the second type plays dead until the trigger starts the malicious code. In simpler words, the infected program or the virus software needs to be executed.

An efficient computer virus must contain a certain search routine that locates new files or new disks which are useful targets for infection. Furthermore, every computer virus must contain a routine to copy itself into the program which the search routine locates. The three main virus parts are:

Infection mechanism

1. Infection mechanism is also known as an infection vector. Using this mechanism the virus spreads or propagates. Every virus usually has a search routine, which locates new files or new disks for infection.

- **Trigger**

1. The trigger also known as a logic bomb is the compiled version and can be activated any time within an executable file. When the virus is executed, it determines the event or condition for the malicious payload to be activated or delivered which can include a specific date, a particular time, particular occurrence of another program, capacity of the disk beyond some limit, a double-click that opens a particular file and many more.

- **Payload**

1. The "payload" is the actual body or data that performs the required malicious activity of the virus. Payload activity might be noticeable as it causes the system to slow down, as most of the time the "payload" itself is the destructive activity or sometimes non-destructive but distributive, which is called Virus hoax.

WHAT ARE THE DIFFERENT TYPES OF COMPUTER VIRUSES?

FILE INFECTORS

Some file infector viruses attach themselves to program files, usually selected .com or .EXE files. Some can infect any program for which execution is requested, including .SYS, .OVL, .PRG, and .MNU files. When the program is loaded, the virus is loaded as well. Other file infector viruses arrive as wholly contained programs or scripts sent as an attachment to an email note.

MACRO VIRUSES

These viruses specifically target macro language commands in applications like Microsoft Word and other programs. In Word, macros are saved sequences for commands or keystrokes that are embedded in the documents. Macro viruses can add their malicious code to the legitimate macro sequences in a Word file. Microsoft disabled macros by default in more recent versions of Word; as a result, hackers have used social engineering schemes to convince targeted users to enable macros and launch the virus. As macro viruses have seen resurgence in recent years, Microsoft added a new feature in Office 2016 that allows security managers to selectively enable macro use for trusted workflows only, as well as block macros across an organization.

OVERWRITE VIRUSES

Some viruses are designed specifically to destroy a file or application's data. After infecting a system, an overwrite virus begins overwriting files with its own code. These viruses can target specific files or applications or systematically overwrite all files on an infected device. An overwrite virus can install new code in files and applications that programs them to spread the virus to additional files, applications and systems.

POLYMORPHIC VIRUSES

A polymorphic virus is a type of malware that has the ability to change or mutate its underlying code without changing its basic functions or features. This process helps a virus evade detection from many antimalware and threat detection products that rely on identifying signatures of malware; once a polymorphic virus' signature is identified by a security product, the virus can then alter itself so that it will no longer be detected using that signature.

RESIDENT VIRUSES

This type of virus embeds itself in the memory of a system. The original virus program isn't needed to infect new files or applications; even if the original virus is deleted, the version stored in memory can be activated when the operating system loads a specific application or function. Resident viruses are problematic because they can evade antivirus and antimalware software by hiding in the system's RAM.

ROOTKIT VIRUSES

A rootkit virus is a type of malware that installs an unauthorized rootkit on an infected system, giving attackers full control of the system with the ability to fundamentally modify or disable functions and programs. Rootkit viruses were designed to bypass antivirus software, which typically scanned only applications and files. More recent versions of major antivirus and antimalware programs include rootkit scanning to identify and mitigate these types of viruses.

SYSTEM RECORD INFECTORS

These viruses infect executable code found in certain system areas on a disk. They attach to the DOS boot sector on diskettes and USB thumb drives or the Master Boot Record on hard disks. In a typical attack scenario, the victim receives storage device that contains a boot disk virus. When the victim's operating system is running, files on the external storage device can infect the system; rebooting the system will trigger the boot disk virus. An infected storage device connected to a computer can modify or even replace the existing boot code on the infected system so that when the system is booted next, the virus will be loaded and run immediately as part of the master boot record. Boot viruses are less common now as today's devices rely less on physical storage media.

BOOT SECTOR VIRUS

This type of virus can take control when you start or boot the computer. One way it can spread is by plugging an infected USB drive into your computer.

WEB SCRIPTING VIRUS

This type of virus exploits the code of web browsers and web pages. If you access such a web page, the virus can infect your computer.

BROWSER HIJACKER

This type of virus “hijacks” certain web browser functions, and you may be automatically directed to an unintended website.

DIRECT ACTION VIRUS

This type of virus comes into action when you execute a file containing a virus. Otherwise, it remains dormant.

MULTIPARTITE VIRUS

This kind of virus infects and spreads in multiple ways. It can infect both program files and system sectors.

WHERE ARE THE VIRUS RISKS?

A few vulnerable ways that the user might install or run a virus program unknowingly are:

EMAIL can include infected attachments. If the user, double-click on an infected attachment, the user might risk infecting the system. Some emails even include malicious scripts that run as soon as the mail is previewed or read the body text.

FLOPPY DISKS AND CDS can have a virus in the boot sector. They can also hold infected programs or documents. CDs may also hold infected items.

DOCUMENTS AND SPREADSHEETS can contain macro viruses, which can infect and make changes to other documents or spreadsheets.

PROGRAMS that carry a virus can infect your machine as soon as you run them

THE INTERNET DOWNLOADED PROGRAMS or documents may be infected.

SIGNS OF VIRUS INFECTION

It is vital for any computer user to be aware of these warning signs:-

- Slower system performance: It can be observed that the system's efficiency reduces as its speed reduces.
- Pop-ups bombarding the screen: The user can observe that unusual pop up ads will be displayed on the screen and these annoying pop up ads cannot be disabled.
- Programs running on their own: The unwanted programs get executed or starts running on its own even if the user did not want it to.
- Files multiplying/duplicating on their own: The existing files duplicate themselves and therefore take up the system's memory and decrease its speed.
- New files or programs in the computer: The user can find new programs and application installed on the system without the user actually installing them.
- Files, folders or programs getting deleted or corrupted: A few files might get deleted or modified automatically due to these viruses and therefore corrupt them.

If the come across any of the above mentioned signs then there are chances that the computer is infected by a virus or malware.

ANTIVIRUS

Anti-virus software is a software utility that detects, prevents, and removes viruses, worms, and other malware from a computer. Most anti-virus programs include an auto-update feature that permits the program to download profiles of new viruses, enabling the system to check for new threats. Antivirus programs are essential utilities for any computer but the choice of which one is very important. One AV program might find a certain virus or worm while another cannot, or vice-versa. Anti-virus software is also known as an anti-virus program or a vaccine. Antivirus software is a computer program that identifies and removes the computer virus and other malicious software like worms and Trojans from an infected computer. Not only this, antivirus software also protects the computer from further virus attacks. Anti-virus system detects viruses from system like SVCHOST.EXE, SERVICEMGR.EXE, LSASS.EXE, storevirus generated by AUTORUN.INF. Generally Antivirus first check the size & according to it if match the size with its data base then it find out the pattern from that file if so then it will delete it.

FEATURES OF AN EFFECTIVE ANTIVIRUS

REAL-TIME SCANNING

While all antivirus software is specifically designed to detect the presence of malware, not all of them detect in the same way. Ineffective products force you to run a manual scan to determine if any systems have been affected, while the best forms of software have dynamic scanning features that are repeatedly checking your computer for the presence of malicious entities. Without this feature, it's much easier for something to infiltrate your computer and begin causing damage before you even realize it.

AUTOMATIC UPDATES

Updates are vital for all forms of software, but this is especially true when it comes to antivirus. Because new types of malware are constantly being developed, antivirus software needs frequent updates in order to track and contain new threats that didn't even exist when it was first installed. If you have to install updates manually, you might miss

important new protections and expose your system to infection, so always make sure your antivirus software is capable of installing updates automatically and frequently.

PROTECTION FOR MULTIPLE APPS

Threats exist across the entire spectrum of apps and services that you rely on for your everyday tasks. From email clients to instant messenger platforms and certainly internet browsers, harmful software can sneak into your system from a variety of different sources. Antivirus programs need to protect multiple vulnerable apps from potential dangers; otherwise you're leaving your hardware dangerously exposed.

AUTO-CLEAN

If the antivirus software immediately detects malicious software, why wouldn't it delete the code on the spot? Unfortunately, some solutions simply place the malware in a quarantine zone upon detection, waiting for the user to log on and manually delete it. Since there's no reason to leave potentially harmful software on your system, you should choose a program that utilizes an auto-clean feature to rid itself of viruses.

FIGHTS AGAINST ALL TYPES OF MALWARE

Between Trojans, bots, spyware, viruses, etc., there are many different types of malware that can harm your computer, and antivirus programs are sometimes designed only to target a specific type of software. It's better to go with a program that can comprehensively detect all or almost all of the various forms that malware takes.

Along with the above mentioned features the antivirus software should have the following as well:

Antivirus system is a dedicated, system i-specific. It provides full protection against the standard pc types of virus for files and programs used to store on the system. In antivirus there is automatic virus signature update via the internet. Proactive virus signature updates via the network for internet isolated. Antivirus can scan the entire libraries. Antivirus support definition of automatic, pre scheduled periodic scans.

TYPES OF ANTIVIRUS

Antivirus software is distributed in a number of forms, including stand-alone antivirus scanners and internet security suites that offer antivirus protection, along with firewalls, privacy controls and other security protections.

While some operating systems are targeted more frequently by virus developers, antivirus software is available for most OSes:

WINDOWS ANTIVIRUS SOFTWARE

Most antivirus software vendors offer several levels of Windows products at different price points, starting with free versions offering only basic protection. Users must start scans and updates manually and typically free versions of antivirus software won't protect against links to malicious websites or malicious attachments in emails. Premium versions of antivirus software often include suites of endpoint security tools that may provide secure online storage, ad blockers and file encryption. Since 2004, Microsoft has been offering some kind of free antivirus software as part of the Windows operating system itself, generally under the name Windows Defender, though the software was mostly limited to detecting spyware prior to 2006.

MAC OS ANTIVIRUS SOFTWARE

Although macOS viruses exist, they're less common than Windows viruses, so antivirus products for macOS are less standardized than those for Windows. There are a number of free and paid products available, providing on-demand tools to protect against potential malware threats through full-system malware scans and the ability to sift through specific email threads, attachments and various web activities.

ANDROID ANTIVIRUS SOFTWARE

Android is the world's most popular mobile operating system and is installed on more mobile devices than any other OS. Because most mobile malware targets Android, experts recommend all Android device users install antivirus software on their devices. Vendors offer a variety of basic free and paid premium versions of their Android antivirus software including anti-theft and remote-locating features. Some run automatic scans and actively try to stop malicious web pages and files from being opened or downloaded.

VIRUS DETECTION TECHNIQUES

Various virus detection techniques have been developed over the years, in order to keep up with the different types of known and unknown viruses. Most of these techniques fall under one of the four generalized techniques outlined below.

GENERIC SIGNATURE SCANNING

Since static signature scanning method can easily be evaded by modifying certain instructions or rearranging the virus code, therefore generic signature or scanning using wildcards is also used by antivirus scanners to detect viruses. In this method the virus signatures are defined using wildcards which allows the scanner to skip certain bytes and detect all variants of the same virus family. This method works in the same way as regular expressions are used to define and recognize a pattern from the long string. This method is effective in against viruses which may mutate their code slightly or rearrange certain instructions to evade detection. Also since many new viruses are also created by modifying code of previous existing viruses, using wildcard scanning may also help in detecting these new variants.

For example, 0400 B801 020E 07BB??02 %3 33C9 8BD1 419C. The ‘??’ denotes the wildcard characters which can be skipped by the scanner while scanning the input file.

HEURISTIC ANALYSIS

Several types of viruses such as encryptor, decryptor and metamorphic viruses cannot be detected using signature matching techniques, as the virus signature can keep changing, or become unreadable due to encryption. In such cases, the heuristic analysis technique can be used, which works by observing the behavior of a binary file, either through static analysis of the binary or dynamic analysis, in a virtual environment, and then determining how likely the file is to be a threat. In the static heuristic analysis, the binary file is reverse engineered to obtain the code. The code is then analyzed and fragments of code that are likely to cause suspicious behavior are identified, by comparing with a database of code fragments that cause harmful effects. Static heuristic analysis is different from signature matching. In signature matching we match the code of the executable with a database of known viruses, and a perfect match gives us the exact name and family of the virus with complete certainty. In static heuristic analysis, we do not test against known

virus signatures, but against code fragments which are likely to cause virus like behavior such as replication, metamorphosis, deleting files etc. If the file is likely enough to exhibit virus like behavior, it is flagged and reported. In dynamic heuristic analysis, a virtual environment is created and the file is executed inside this virtual environment. It is allowed to function as it would normally, and its behavior is observed in this isolated environment. If the file performs any suspicious actions, such as adding questionable registry entries, replication to other files, formatting the hard disk and such, the file is flagged as a virus and reported. This method is slower than the static analysis, but more likely to detect an unknown virus and less susceptible to false positives, as the file is allowed to run its course of execution and its exact behavior is observed. Thus, heuristic analysis makes it possible to probabilistically detect new and unknown viruses, whose exact signatures are not present in the signatures database. Heuristic analysis is required in addition to signature matching not only to detect new virus families, but also to detect self-mutating and encrypted viruses.

STATIC SIGNATURE-BASED METHOD (STRING SCANNING)

This is the most effective and common way to identify known viruses. The virus is simply a piece of unwanted code which gets attached to a file. This code will consist of series of instructions for the computer to execute in order for the virus to function. These instructions must be present in exactly the same order in each of the infected file for the virus to do its job. These specific set of instructions are unique to a specific virus and are unlikely to be found anywhere else in a normal program. These instructions or strings of bytes are referred to as “virus signatures”. Most of the antivirus programs look for these virus signatures or strings of bytes in the file they are analyzing for threat. If the virus signature is found in a particular file or target program being analyzed, it is marked as infected. For the antivirus program to be able to recognize a virus, its virus signature must be present in its signature database. The efficiency and effectiveness of an antivirus program depends on its ability to search and match the presence of virus signature in the file from the thousands of virus signatures in its database. Increase in the number of virus signatures in the antivirus database increases the ability of the antivirus to detect different types of viruses. If a particular virus signature is not present in the antivirus database, then that virus cannot be detected by the antivirus program.

For example, if a file contains the string 0400 B801 020E 07BB 0002 33C9 8BD1 419C then the file is infected with Stoned virus.

The above mentioned system uses STATIC SIGNATURE-BASED DETECTION METHOD to check if a file or a directory is a virus or a safe program. In this method the binary code of a file scanned for virus signatures. If the virus signatures are present, the file is a virus file else it is a safe file.

HOW DOES AN ANTIVIRUS WORK?

An antivirus program works by scanning files, directories or the whole system/device itself for malicious files or programs. Any malicious code that's detected would be notified to the user and when asked to clean, the antivirus would clean it. Thus, on the one hand, the antivirus software keeps the system free of malware and thus prevents data theft and other serious damages, while on the other hand, it prevents the system from slowing down due to malware infection. Antivirus software thus improves system performance on the whole.

Let us consider a virus file as shown:

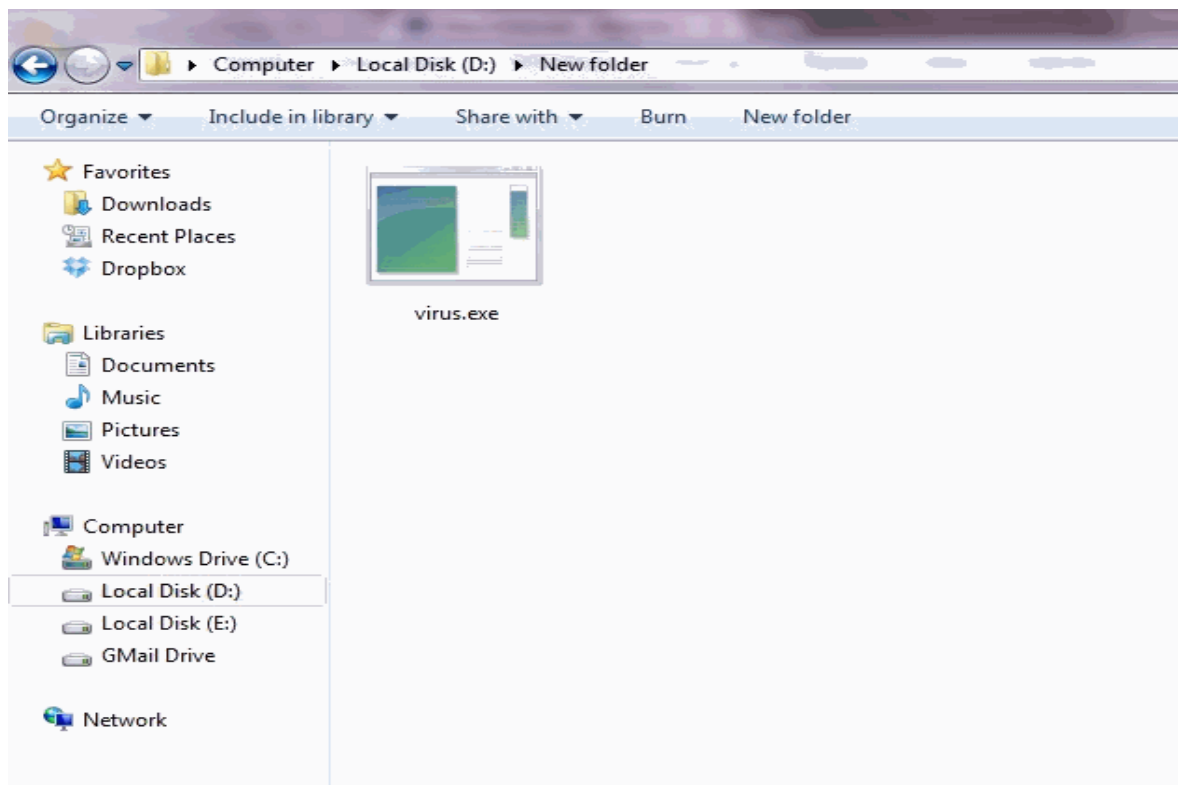


Fig 1.1 AN EXECUTABLE VIRUS FILE INSTALLED IN THE SYSTEM

The above figure shows the virus.exe installed in a system. The Binary code of this file can be viewed by opening this file in Notepad++, a text editor.

Upon opening the file, we will see all unknown characters in file.

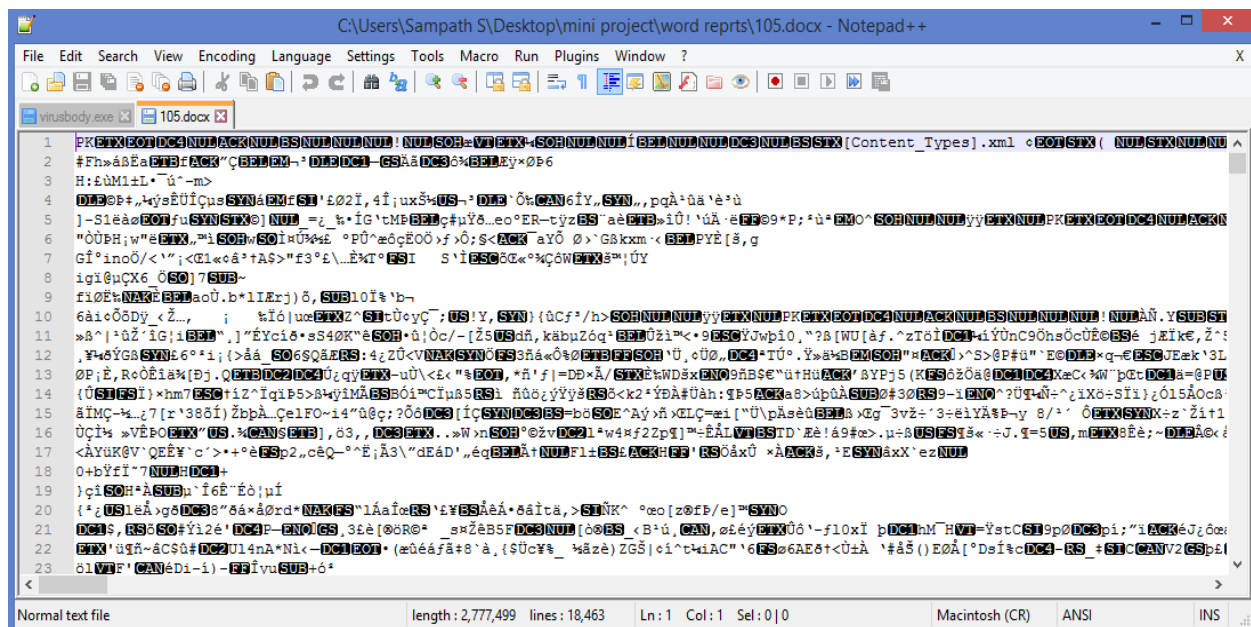


Fig 1.2 BINARY CODE OF A NORMAL FILE

The above figure shows the binary code of a safe file that is not infected by virus. The

above binary code has no virus signatures.



Fig 1.3 VIRUS SIGNATURES OF A VIRUS FILE

The above figure depicts the binary code of a virus infected file. The virus signatures present in the above file helps the antivirus software to detect virus.

Table 1.1 IDENTIFIABLE CHARACTERS IN A VIRUS FILE

Identifiable Characters in Virus.Exe File			
Sr. No.	Character	Line No.	Character No.
1	M	1	1
2	Z	1	2
3	P	2	9
4	E	2	10
5	(9	3
6	%	9	4

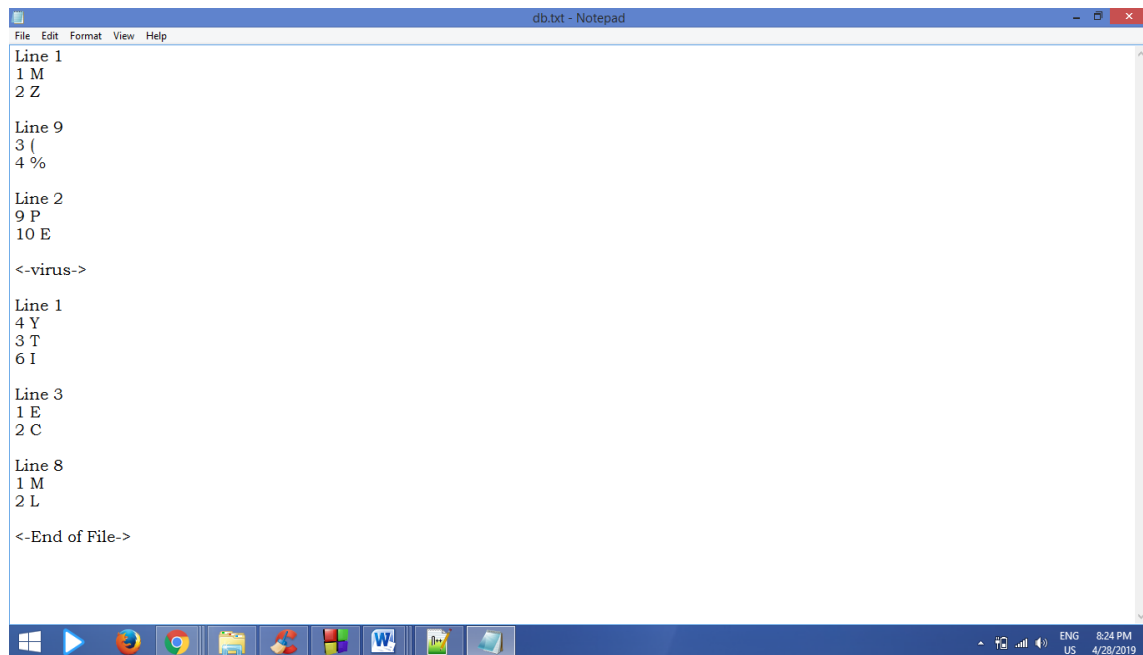


Fig 1.4 VIRUS DATABASE THAT CONTAINS THE VIRUS SIGNATURES

The antivirus program compares the binary code of a file with the virus database and searches for the signatures saved in the virus database.

If the required signatures are found then the antivirus program deletes the particular file as the file is a virus file.

1.1 Motivation of project

The point of malware is to infect the computer in order to search for something the attacker can use, such as personal information like banking account names, passwords, or control over your computer.

Protecting our personal information is a no-brainer because if a criminal has our personal information, it's easy for them to change passwords and email addresses, and then they can start wiring money from your accounts to their own. Although not every intrusion into a computer is meant to cause damage or steal valuable information, that doesn't mean that the attack isn't dangerous. All intrusions into a computer exploit what is known as vulnerability, or a weakness in the computer's operating system or other software that can act as an access point to an attack. Once even the most innocuous of an intrusion exploits vulnerability, it basically sends a signal to others that this computer has been infiltrated. This opens the door wide open to much worse attacks.

The malware for these purposes usually comes in the form of Trojans that piggyback on a document or program that you install. This program-within-a-program could include a key logger that records what you type, which is then sent to the attacker.

If a virus takes over our computer and makes it unusable for us, it could be because the attacker is using it remotely. The attacker could be sifting through our files for passwords and account information, but they could use our computer to perform attacks on bigger fish. For example, many attackers use viruses to infiltrate several anonymous computers to perform distributed denial of service (DDoS) attacks to bring down a website or a network. The benefit for the attacker is that it's much harder for the main target to pinpoint where the attacks are coming from if they're using multiple computers, including yours. That makes it harder to stop the attacks, and the attacker gains the benefit of anonymity.

For as long as computers have been and will be in existence, whether connected to the Internet or not, there will always be a need for antivirus software. There will never be a time when people, whether mischievous youths seeking a thrill or a hardened cybercriminals looking to exploit billion-dollar companies, will stop looking to find ways to commit fraud, cause widespread damage, or just experience the rush of breaking into a computer.

Antivirus software is an important tool to help prevent such attacks. Not every type of cyberattack can be prevented with antivirus software, but it can be a great asset when trying to prevent intrusion into a computer.

Although not every intrusion into a computer is meant to cause damage or steal valuable information, that doesn't mean that the attack isn't dangerous. All intrusions into a computer exploit what is known as vulnerability, or a weakness in the computer's operating system or other software that can act as an access point to an attack. Once even the most innocuous of an intrusion exploits vulnerability, it basically sends a signal to others that this computer has been infiltrated. This opens the door wide open to much worse attacks.

When looking to purchase antivirus software, make sure to purchase a trusted and well known, subscription-based program. There will never be a time when people, whether mischievous youths seeking a thrill or a hardened cybercriminals looking to exploit billion-dollar companies, will stop looking to find ways to commit fraud, cause widespread damage, or just experience the rush of breaking into a computer.

This is important, as the makers of this type of software will be able to keep their subscribers' computers protected with real-time updates that scout out the latest threats.

Not having anti-virus on a computer is like inviting a criminal into the home or having an uninvited guest! They then cause havoc or steal from the owner. Today's internet has provided many ways for virus attacks and there are thousands of threats. To be safe from these it is vital to police the computer and have it protected at all times. The Importance Of Antivirus Software cannot be underestimated.

1.2 Problem statement

To design and implement a antivirus program using C++ to scan and detect virus files and delete them so as ensure that the computer is protected from virus.

Chapter 02

SYSTEM REQUIREMENT SPECIFICATION

Purpose:

The main purpose of the above proposed system is that it reduces the risk faced due to intrusion of virus that can reduce the efficiency of a system. Through the above proposed system the scanning and detection of virus can be made simpler, less time consuming and effective.

The purposes of implementing an antivirus using C++ are:

1. To implement different constructs of C++ language like loops, branching constructs, switch cases and many more.
2. To be able to solve different problem statements in C++.
3. To be able to work effectively and hence develop technical skills.
4. To understand different concept of C++.

The main objectives of antivirus in C++ are:

- Scan specific files or directories for any malware or known malicious patterns
- Allow you to schedule scans to automatically run for you
- Allow you to initiate a scan of a particular file or your entire computer, or of a CD or flash drive at any time.
- Remove any malicious code detected –sometimes you will be notified of an infection and asked if you want to clean the file, other programs will automatically do this behind the scenes.
- Show you the ‘health’ of your computer

Scope

- Ensures the safety of a computer.
- A file or a directory can be scanned anytime anywhere.
- Less time consuming as the entire system is not scanned.
- Deletes the virus file automatically

2.1 Hardware System Configuration

Processor -Intel Core i5

Speed -1.8 GHz

RAM -256 MB (min)

Hard disk -10 GB

2.2 Software System Configuration

Operating System -Windows 8

Programming Language –C++ language

Compiler -Code:: blocks

2.3 About the Language

C++ is a statically typed, compiled, general-purpose, case-sensitive, free-form programming language that supports procedural, object-oriented, and generic programming. C++ is regarded as a middle-level language, as it comprises a combination of both high-level and low-level language features. C++ was developed by Bjarne Stroustrup starting in 1979 at Bell Labs in Murray Hill, New Jersey, as an enhancement to the C language and originally named C with Classes but later it was renamed C++ in 1983. C++ is a superset of C, and that virtually any legal C program is a legal C++ program.

C++ fully supports object-oriented programming, including the four pillars of object-oriented development:

1. Encapsulation
2. Data hiding
3. Inheritance
4. Polymorphism

Standard C++ consists of three important parts:

- The core language giving all the building blocks including variables, data types and literals, etc.
- The C++ Standard Library giving a rich set of functions manipulating files, strings, etc.
- The Standard Template Library (STL) giving a rich set of methods manipulating data structures, etc.

THE ANSI STANDARD

The ANSI standard is an attempt to ensure that C++ is portable; that code you write for Microsoft's compiler will compile without errors, using a compiler on a Mac, UNIX, a Windows box, or an Alpha.

When we consider a C++ program, it can be defined as a collection of objects that communicate via invoking each other's methods. Let us now briefly look into what a class, object, methods, and instant variables mean.

- Object - Objects have states and behaviors. Example: A dog has states - color, name, breed as well as behaviors - wagging, barking, and eating. An object is an instance of a class.
- Class - A class can be defined as a template/blueprint that describes the behaviors/states that object of its type support.
- Methods - A method is basically a behavior. A class can contain many methods. It is in methods where the logics are written, data is manipulated and all the actions are executed.
- Instant Variables - Each object has its unique set of instant variables. An object's state is created by the values assigned to these instant variables.

STRUCTURE

- The C++ language defines several headers, which contain information that is either necessary or useful to your program. For all the programs, the header `#include<iostream>` is needed.

- The line using namespace std; tells the compiler to use the std namespace. Namespaces are a relatively recent addition to C++.
- The next line '// main()' is where program execution begins.' is a single-line comment available in C++. Single-line comments begin with // and stop at the end of the line.
- The line int main() is the main function where program execution begins.
- The cout and cin objects are used to print the output on the screen and extract the input from the user respectively.
- The next line return 0; terminates main() function and causes it to return the value 0 to the calling process.

DATA TYPES

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory.

PRIMITIVE BUILT-IN TYPES

C++ offers the programmer a rich assortment of built-in as well as user defined data types. Following table lists down seven basic C++ data types:

Table 2.1 BUILT-IN DATA TYPES

Type	Keyword
Boolean	bool
Character	char
Integer	int
Floating point	float
Double floating point	double
Valueless	void
Wide character	wchar_t

Several of the basic types can be modified using one or more of these type modifiers:

- signed
- unsigned
- short
- long

The modifiers signed, unsigned, long, and short can be applied to integer base types. In addition, signed and unsigned can be applied to char, and long can be applied to double. The modifiers signed and unsigned can also be used as prefix to long or short modifiers. For example, unsigned long int. C++ allows a shorthand notation for declaring unsigned, short, or long integers. You can simply use the word unsigned, short, or long, without int. It automatically implies int. For example, the following two statements both declare unsigned integer variables.

The following table shows the variable type, how much memory it takes to store the value in memory, and what is maximum and minimum value which can be stored in such type of variables.

Table 2.2 SIZES OF VARIABLES

Type	Typical Bit Width	Typical Range
char	1byte	-127 to 127 or 0 to 255
unsigned char	1byte	0 to 255
signed char	1byte	-127 to 127
int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295
signed int	4bytes	-2147483648 to 2147483647
short int	2bytes	-32768 to 32767
unsigned short int	Range	0 to 65,535
signed short int	Range	-32768 to 32767
long int	4bytes	-2,147,483,647 to 2,147,483,647
signed long int	4bytes	same as long int
unsigned long int	4bytes	0 to 4,294,967,295
float	4bytes	+/- 3.4e +/- 38 (~7 digits)
double	8bytes	+/- 1.7e +/- 308 (~15 digits)
long double	8bytes	+/- 1.7e +/- 308 (~15 digits)

SCOPE OF A VARIABLE

A scope is a region of the program and broadly speaking there are three places, where variables can be declared:

- Inside a function or a block which is called local variables
- In the definition of function parameters which is called formal parameters
- Outside of all functions which is called global variables.

STORAGE CLASSES

A storage class defines the scope (visibility) and life-time of variables and functions within a C++ Program. These specifiers precede the type that they modify. There are following storage classes, which can be used in a C++ Program

- Auto
- Register
- Static
- Extern
- Mutable

THE AUTO STORAGE CLASS

The auto storage class is the default storage class for all local variables.

THE REGISTER STORAGE CLASS

The register storage class is used to define local variables that should be stored in a register instead of RAM. This means that the variable has a maximum size equal to the register size and can't have the unary '&' operator applied to it as it does not have a memory location.

The register should only be used for variables that require quick access such as counters. It should also be noted that defining 'register' does not mean that the variable will be stored in a register. It means that it MIGHT be stored in a register depending on hardware and implementation restrictions.

THE STATIC STORAGE CLASS

The static storage class instructs the compiler to keep a local variable in existence during the life-time of the program instead of creating and destroying it each time it comes into and goes out of scope. Therefore, making local variables static allows them to maintain their values between function calls. The static modifier may also be applied to global variables. When this is done, it causes that variable's scope to be restricted to the file in which it is declared. In C++, when static is used on a class data member, it causes only one copy of that member to be shared by all objects of its class.

THE EXTERN STORAGE CLASS

The extern storage class is used to give a reference of a global variable that is visible to ALL the program files. When you use 'extern' the variable cannot be initialized as all it does is point the variable name at a storage location that has been previously defined. When you have multiple files and you define a global variable or function, which will be used in other files also, then extern will be used in another file to give reference of defined variable or function. Just for understanding extern is used to declare a global variable or function in another file.

THE MUTABLE STORAGE CLASS

The mutable specifier applies only to class objects, which are discussed later in this tutorial. It allows a member of an object to override const member function. That is, a mutable member can be modified by a const member function.

LOOP TYPES

There may be a situation, when you need to execute a block of code several numbers of times. In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on. Programming languages provide

various control structures that allow for more complicated execution paths. A loop statement allows us to execute a statement or group of statements multiple times and following is the general form of a loop statement in most of the programming languages:

Table 2.3 LOOP TYPES AND DESCRIPTION

Loop Type	Description
while loop	Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body.
for loop	Execute a sequence of statements multiple times and abbreviates the code that manages the loop variable.
do...while loop	Like a 'while' statement, except that it tests the condition at the end of the loop body.
nested loops	You can use one or more loop inside any another 'while', 'for' or 'do..while' loop.

A **WHILE LOOP** statement repeatedly executes a target statement as long as a given condition is true.

SYNTAX

```
while(condition)
{
    statement(s);
}
```

Here, statement(s) may be a single statement or a block of statements. The condition may be any expression, and true is any non-zero value. The loop iterates while the condition is true. When the condition becomes false, program control passes to the line immediately following the loop.

FOR LOOP is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

SYNTAX

```
for ( init; condition; increment )  
{  
    statement(s);  
}
```

Here is the flow of control in a for loop:

1. The init step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
2. Next, the condition is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and flow of control jumps to the next statement just after the for loop.
3. After the body of the for loop executes, the flow of control jumps back up to the increment statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.
4. The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again condition). After the condition becomes false, the for loop terminates.

DO...WHILE LOOP test the loop condition at the top of the loop, the do...while loop checks its condition at the bottom of the loop. A do...while loop is similar to a while loop, except that a do...while loop is guaranteed to execute at least one time.

SYNTAX

```
do{  
    statement(s);  
}while( condition );
```

Notice that the conditional expression appears at the end of the loop, so the statement(s) in the loop execute once before the condition is tested. If the condition is true, the flow of control jumps back up to do, and the statement(s) in the loop execute again. This process repeats until the given condition becomes false.

THE STANDARD OUTPUT STREAM (COUT)

The predefined object `cout` is an instance of `ostream` class. The `cout` object is said to be "connected to" the standard output device, which usually is the display screen. The `cout` is used in conjunction with the stream insertion operator, which is written as `<<` which are two less than signs.

THE STANDARD INPUT STREAM (CIN)

The predefined object `cin` is an instance of `istream` class. The `cin` object is said to be attached to the standard input device, which usually is the keyboard. The `cin` is used in conjunction with the stream extraction operator, which is written as `>>` which are two greater than signs.

APPLICATION OF C++

C++ is used by hundreds of thousands of programmers in essentially every application domain.

C++ is being highly used to write device drivers and other software that rely on direct manipulation of hardware under real-time constraints.

C++ is widely used for teaching and research because it is clean enough for successful teaching of basic concepts.

Anyone who has used either an Apple Macintosh or a PC running Windows has indirectly used C++ because the primary user interfaces of these systems are written in C++.

Chapter 03

METHODOLOGY

3.1 Flowchart

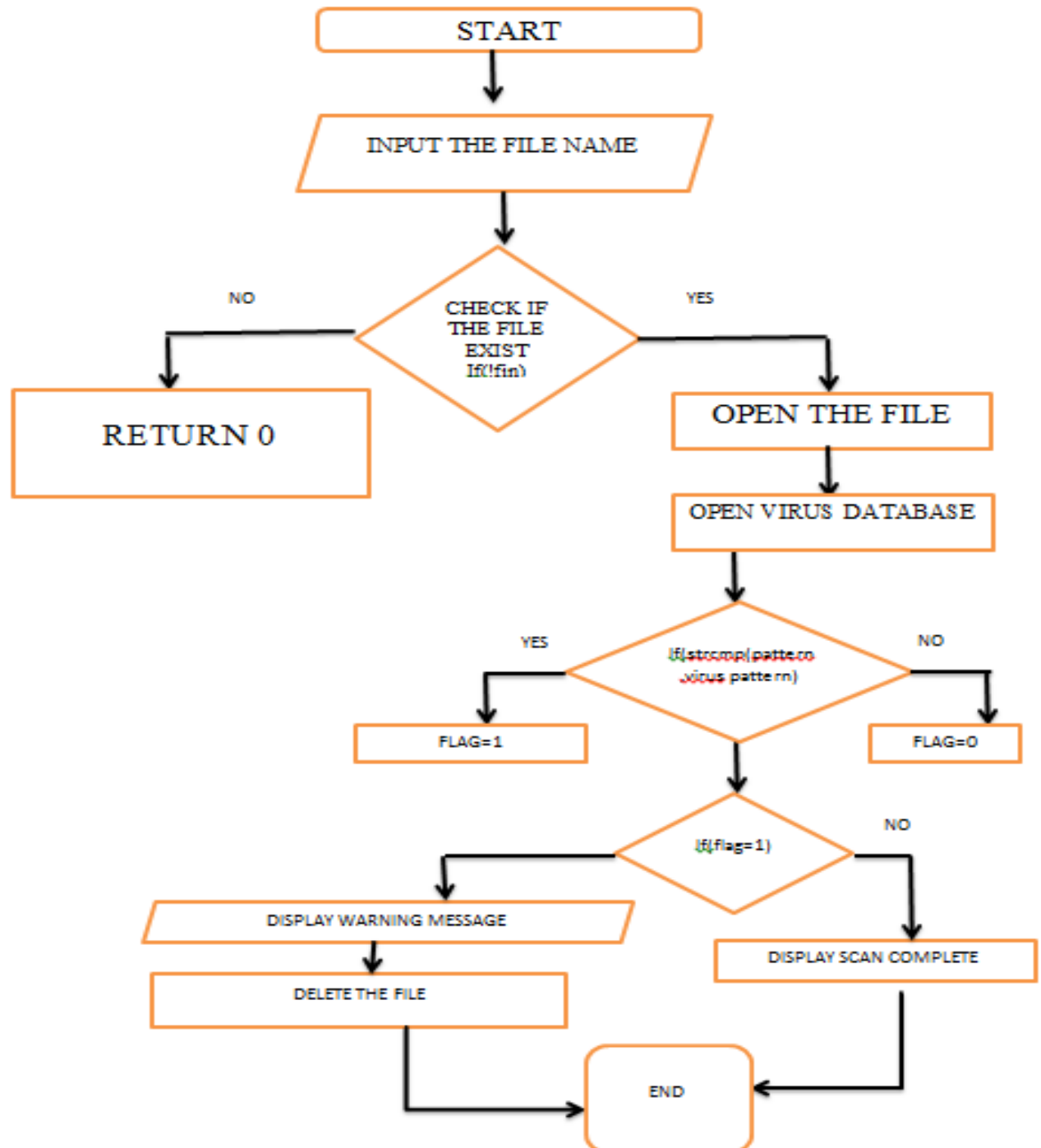


fig 3.1 FLOWCHART FOR IMPLEMENTATION OF ANTIVIRUS USING C++

The above flowchart shows the flow of the program. Once a file name is given as an input, it is checked if it exists. If the file exists, the file is scanned for virus.

3.2 Algorithm

The antivirus code uses the below mentioned algorithm to scan and detect virus. The algorithm goes like:

1. Start
2. Ask the user to enter the directory or the file name
3. Check if the file exists.
 - a. If yes open the file in binary code.
 - i. open the virus database.
 - ii. Compare the patterns in the binary code with that present in the virus database.
 - iii. If(`strcmp(pattern.virus pattern)==1`)
 1. `Cout<<"virus found"<<endl;`
 2. Delete the file
 - iv. Else
 1. `Cout<<"scan complete"<<endl;`
 - b. If no return 0.
4. Close all the files
5. End

3.3 Code and Implementation

```
#include<windows.h>
#include <dirent.h>
#include <string.h>
#include <fstream>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
using namespace std;

int scan_this(char *file_name)
{
    char *pattern, *line_in_file;
    char file_ch, ch;
    int val, val2, flag;
    ifstream fin3, fin4;
    fin3.open(file_name); // incase the file is not accesible
    if(fin3)
    {
        fin4.open("db.txt"); // this is our character pattern file
        for(;;)
        {
            fin4>>pattern;
            if(!strcmp(pattern,"<-"))
            {
                fin4>>pattern;
                if(!strcmpi(pattern,"End"))return -1;
                else if(!strcmpi(pattern, "virus"))
                {
                    if(flag) return 1;
                    else continue;
                }
            }
        }
    }
}
```

```
}  
}  
else if(!strcmpi(pattern,"LINE"))  
{  
    fin4>>val; // got the line number  
    for(int i=0;i<val-1;i++)  
    {  
        fin3.getline(line_in_file, 300);  
    }  
    fin4>>val; // got the character number  
    fin4>>file_ch; // got the character  
    //skipping initial character to reach the character  
    for(int i=0;i<val-1;i++)  
    {  
        fin3.get(ch);  
    }  
    if(file_ch == ch) flag = 1; // matched.  
    else flag =0;  
    fin3.seekg(0); // set to start  
}  
}  
}  
}  
  
int main()  
{  
    char comm[300], dirpath[100], file_name[200];  
    char ask;  
    int response;  
    ifstream fin;  
    cout<<"Enter Directory you want to scan: ";  
    cin>>dirpath;  
    strcpy(comm, "dir ");
```

```
strcat(comm, "dirpath /b /s >tmp.$$$");
system(comm);
fin.open("tmp.$$$");
while(!fin.eof())
{
    fin.getline(file_name, 200);
    response = scan_this(file_name);
    if(response == 1)
    {
        cout<<"<-!! Caution.! A Virus has been Detected..!";
        cout<<"n"<<file_name;
        cout<<"Press Enter Key to Delete it.";
        ask= getch();
        if(ask == 13)
        {
            remove(file_name); // delete the virus
        }
    }
}
fin.close();
cout<<"Scan Complete!! Thank You for using our anti virus";
getch();
}
```

Chapter 04

RESULTS AND DISCUSSION

4.1 Output snapshots

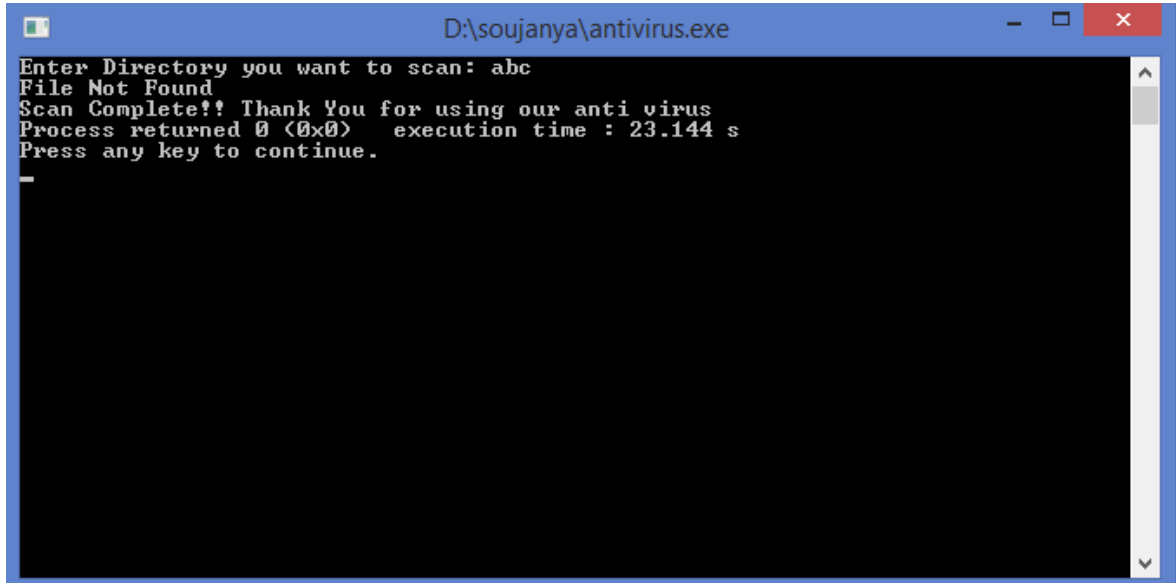
A screenshot of a Windows command prompt window titled "D:\soujanya\antivirus.exe". The text inside the window reads: "Enter Directory you want to scan: abc", "File Not Found", "Scan Complete!! Thank You for using our anti virus", "Process returned 0 (0x0) execution time : 23.144 s", and "Press any key to continue.". The window has a blue title bar and standard Windows window controls (minimize, maximize, close) in the top right corner.

Fig 4.1 WHEN THE FILE ENTERED IS NOT FOUND

When the entered file name is not found in the system the program displays an “file not found” message.

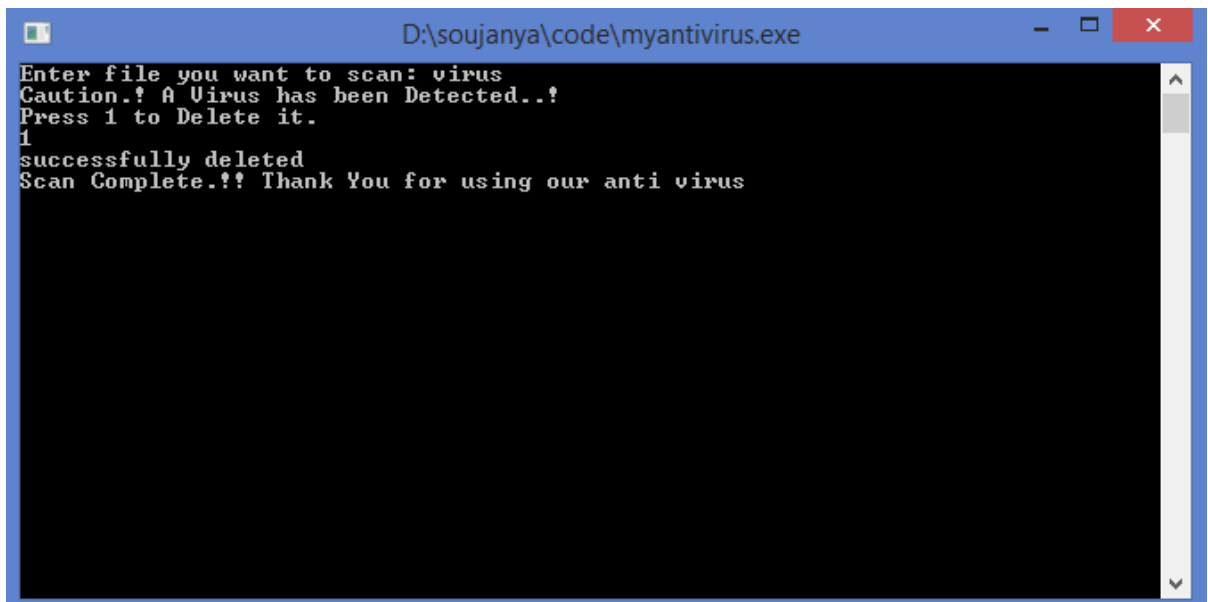
A screenshot of a Windows command prompt window titled "D:\soujanya\code\myantivirus.exe". The text inside the window reads: "Enter file you want to scan: virus", "Caution..! A Virus has been Detected..!", "Press 1 to Delete it.", "1", "successfully deleted", and "Scan Complete..!! Thank You for using our anti virus". The window has a blue title bar and standard Windows window controls (minimize, maximize, close) in the top right corner.

Fig 4.2 A VIRUS FILE IS DETECTED AND DELETED

Here the file named virus is a virus infected file. Therefore when this file is scanned a “caution” is displayed and deleted from the system.

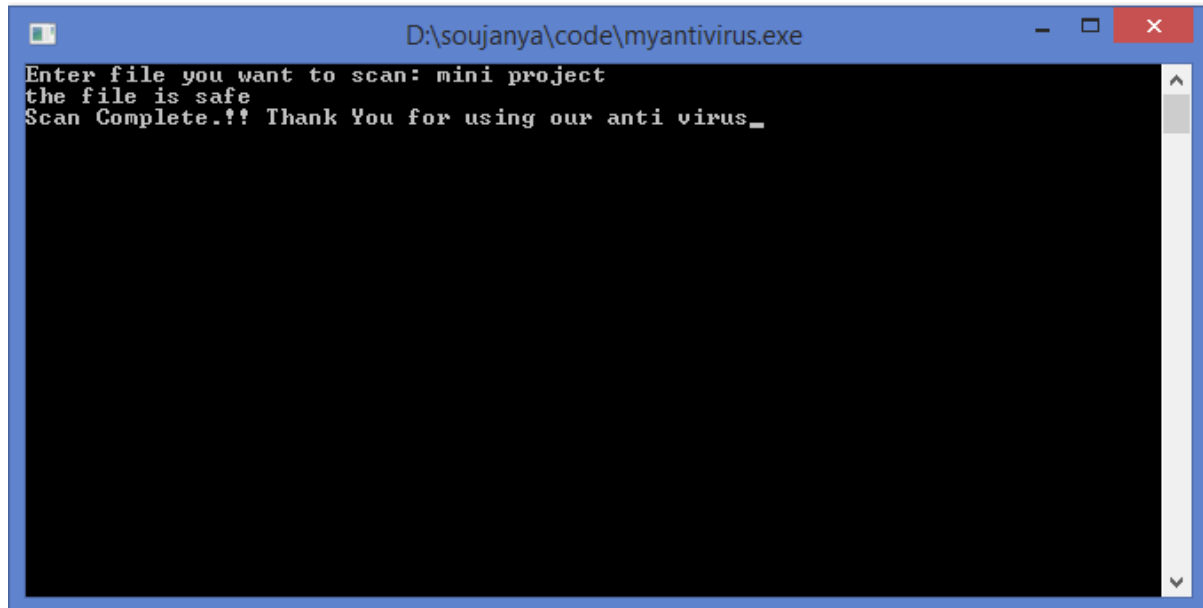


Fig 4.3 A VIRUS IS NOT DETECTED; THEREFORE FILE SAFE MESSAGE IS
DISPLAYED.

The entered filename is safe from viruses. Therefore the “file is safe message” is displayed.

CONCLUSION AND FUTURE ENHANCEMENTS

There are plenty of free antivirus software programs that work well at detecting, preventing and removing malware, but they may be lacking some more advanced features. When you pay for antivirus software, you're paying for the latest updated information and better detection. For example, the best antivirus software developers apply heuristics to find new viruses that could pass undetected through a free version. The attackers who are writing code for those viruses are smart –of course they're going to pass their virus through the most common free versions of antivirus software before launching it at unwitting victims.

Future antimalware software solutions will be based upon a distributed and multi-tiered deployment from the "cloud" to the endpoint. This means that telemetry and behavioral analytics are exchanged between all elements of the collective "network" of communities of interest to provide both detective and preventative capabilities. Intelligent monitoring and correlation across not only antimalware software platforms, but any and all networked elements, will be critical in the evolution of detecting, isolating and mitigating the onslaught. Further, as attacks become more targeted and focused on not only the exploitation but also the extraction of information, decisions on content in context will also be required.

REFERENCES

- [1] <https://freaksense.com/how-to-make-antivirus-using-c-programming-language/>
- [2] <https://www.scribd.com/doc/103002323/Seminar-Report-on-Antivirus>
- [3] <https://srbunty.wordpress.com/2013/06/24/make-antivirus-using-c-programming-language/>
- [4] http://cds.iisc.ac.in/wp-content/uploads/DS286.AUG2016.Lab2_.cpp_tutorial.pdf
- [5] <https://en.wikipedia.org/wiki/C%2B%2B>
- [6] https://www.google.com/search?q=scope+of+antivirus&rlz=1C1DFOC_enIN610IN610&oq=sco&aqs=chrome.1.69i57j69i59j69i65j69i60j69i61j69i60.17671j0j8&sourceid=chrome&ie=UTF-8
- [7] <https://www.toptenreviews.com/software/articles/what-is-antivirus-software/>
- [8] <https://www.webroot.com/in/en/resources/tips-articles/what-is-anti-virus-software>
- [9] <https://searchsecurity.techtarget.com/definition/antivirus-software>