Free and Open Source Software Journal

Index

No	Topic	Date	Sign
1	Identify any Open Source software and create detailed report about it.		
2	Learn at least three different open source licenses and create a brief report about them.		
3	Contributing to Open Source Wikipedia:		
4	GitHub : Open source software		
5	Hands on with Open Source Software- GCC		
6	Hands on with open Source Software- Presentation		
7	Virtualization		
8	Containerization		

Practical 1

Linux

(A) Idea

Linux is a family of free and open-source software operating systems built around the Linux kernel. Typically, Linux is packaged in a form known as a Linux distribution (or distro for short) for both desktop and server use. The defining component of a Linux distribution is the Linux kernel, an operating system kernel first released on September 17, 1991, by Linus Torvalds.

Linux was originally developed for personal computers based on the Intel x86 architecture, but has since been ported to more platforms than any other operating system. Because of the dominance of the Linux kernel-based Android OS on smartphones, Linux has the largest installed base of all general-purpose operating systems.

(B) What problem does it solve

Linux vendors and communities combine and distribute the kernel, GNU components, and non-GNU components, with additional package management software in the form of Linux distributions.

Torvalds continues to direct the development of the kernel. Stallman heads the Free Software Foundation, which in turn supports the GNU components. Finally, individuals and corporations develop third-party non-GNU components. These third-party components comprise a vast body of work and may include both kernel modules and user applications and libraries.

(C) Licensing model

Linux was originally developed as a free Operating System (OS) for Intel x86-based Personal Computers. Since then, the system has been ported to more computer hardware platforms than any other OS. Linux is a leading OS on servers, mainframe computers and supercomputers. As of June 2013, more than 95 percent of the world's 500 fastest supercomputers run some variant of Linux, including the top 80.

Linux kernel is licensed under the GNU General Public License (GPL), version 2. The GPL requires that anyone who distributes software based on source code under this license, must make the originating source code (and any modifications) available to the recipient under the same terms. Other key components of a typical Linux distribution are also mainly licensed under the GPL, but they may use other licenses; many libraries use the GNU Lesser General Public License (LGPL), a more permissive variant of the GPL, and the X.Org implementation of the X Window System uses the MIT License.

The Free Software Foundation uses the name GNU/Linux to refer to the operating system family, as well as specific distributions, to emphasize that most Linux distributions are not just the Linux kernel, and that they have in common not only the kernel, but also numerous utilities and libraries, a large proportion of which are from the GNU project. This has led to some controversy.

(D) Intent behind making it an open source software

The Free Software Foundation uses the name GNU/Linux to refer to the operating system family, as well as specific distributions, to emphasize that most Linux distributions are not just the Linux kernel, and that they have in common not only the kernel, but also numerous utilities and libraries, a large proportion of which are from the GNU project. This has led to some controversy.

Linux was originally developed for personal computers based on the Intel x86 architecture, but has since been ported to more platforms than any other operating system. The development of Linux is one of the most prominent examples of free and open-source software collaboration.

(E) Monetization model

A 2001 study of Red Hat Linux 7.1 found that this distribution contained 30 million source lines of code. Using the Constructive Cost Model, the study estimated that this distribution required about eight thousand person-years of development time. According to the study, if all this software had been developed by conventional proprietary means, it would have cost about $1.53 billion (2018 US dollars) to develop in the United States. Most of the source code (71%) was written in the C programming language, but many other languages were used, including C++, Lisp, assembly language, Perl, Python, Fortran, and various shell scripting languages. Slightly over half of all lines of code were licensed under the GPL. The Linux kernel itself was 2.4 million lines of code, or 8% of the total.

In a later study, the same analysis was performed for Debian version 4.0 (etch, which was released in 2007). This distribution contained close to 283 million source lines of code, and the study estimated that it would have required about seventy three thousand man-years and cost US$8.46 billion (in 2018 dollars) to develop by conventional means.

(F) Popularity

Some of the most popular and mainstream Linux distributions are Arch Linux, CentOS, Debian and Raspbian, Fedora, Gentoo Linux, Linux Mint, Mageia, openSUSE and Ubuntu, together with commercial distributions such as Red Hat Enterprise Linux and SUSE Linux Enterprise Server. Distributions include the Linux kernel, supporting utilities and libraries, many of which are provided by the GNU Project, and usually a large amount of application software to fulfil the distribution's intended use. Desktop Linux distributions include a windowing system, such as X11, Mir or a Wayland implementation, and an accompanying desktop environment such as

GNOME or KDE Plasma; some distributions may also include a less resource-intensive desktop, such as LXDE or Xfce. Distributions intended to run on servers may omit all graphical environments from the standard install, and instead include other software to set up and operate a solution stack such as LAMP. Because Linux is freely redistributable, anyone may create a distribution for any intended use.

Besides the Linux distributions designed for general-purpose use on desktops and servers, distributions may be specialized for different purposes including: computer architecture support, embedded systems, stability, security, localization to a specific region or language, targeting of specific user groups, support for real-time applications, or commitment to a given desktop environment. Furthermore, some distributions deliberately include only free software. As of 2015, over four hundred Linux distributions are actively developed, with about a dozen distributions being most popular for general-purpose use.

(G) Impact

The development of Linux is one of the most prominent examples of free and open-source software collaboration. The underlying source code may be used, modified and distributed—commercially or non-commercially—by anyone under the terms of its respective licenses, such as the GNU General Public License.

Torvalds continues to direct the development of the kernel. Stallman heads the Free Software Foundation, which in turn supports the GNU components. Finally, individuals and corporations develop third-party non-GNU components. These third-party components comprise a vast body of work and may include both kernel modules and user applications and libraries.
Linux vendors and communities combine and distribute the kernel, GNU components, and non-GNU components, with additional package management software in the form of Linux distributions.

Bibliography:

www.google.com

www.wikipedia.org

www.linux.org

www.opensource.com

Practical 2

Apache License:

(A) History of the License

Version 1.1: The Apache License 1.1 was approved by the ASF in 2000: The primary change from the 1.0 license is in the 'advertising clause' (section 3 of the 1.0 license); derived products are no longer required to include attribution in their advertising materials, but only in their documentation.

Version 2.0: The ASF adopted the Apache License 2.0 in January 2004. The stated goals of the license included making the license easier for non-ASF projects to use, improving compatibility with GPL-based software, allowing the license to be included by reference instead of listed in every file, clarifying the license on contributions, and requiring a patent license on contributions that necessarily infringe a contributor's own patents.

(B) Idea of the License

The 1.1 version of the Apache License was approved by the ASF in 2000. The primary change from the 1.0 license is in the 'advertising clause' (section 3 of the 1.0 license); derived products are no longer required to include attribution in their advertising materials, only in their documentation.

Individual packages licensed under the 1.1 version may have used different wording due to varying requirements for attribution or mark identification, but the binding terms were all the same.

The 2.0 version of the Apache License was approved by the ASF in 2004. The goals of this license revision have been to reduce the number of frequently asked questions, to allow the license to be reusable without modification by any project (including non-ASF projects), to allow the license to be included by reference instead of listed in every file, to clarify the license on submission of contributions, to require a patent license on contributions that necessarily infringe the contributor's own patents, and to move comments regarding Apache and other inherited attribution notices to a location outside the license terms (the NOTICE file).

The result is a license that is supposed to be compatible with other open source licenses, while remaining true to the original goals of the Apache Group and supportive of collaborative development across both nonprofit and commercial organizations. The Apache Software Foundation is still trying to determine if this version of the Apache License is compatible with the GPL.

(C) What problem does it solve?

The Apache License is a permissive free software license written by the Apache Software Foundation (ASF).

Like other free software licenses, the license allows the user of the software the freedom to use the software for any purpose, to distribute it, to modify it, and to distribute modified versions of the software, under the terms of the license, without concern for royalties. This makes ALv2 a FRAND-RF license. The ASF and its projects release the software they produce under the Apache License. The license is also used by many non-ASF projects.

(D) Detailed licensing model

The Apache License is permissive in that it does not require a derivative work of the software, or modifications to the original, to be distributed using the same license. It still requires application of the same license to all unmodified parts and, in every licensed file, any original copyright, patent, trademark, and attribution notices in redistributed code must be preserved (excluding notices that do not pertain to any part of the derivative works); and, in every licensed file changed, a notification must be added stating that changes have been made to that file.

If a NOTICE text file is included as part of the distribution of the original work, then derivative works must include a readable copy of these notices within a NOTICE text file distributed as part of the derivative works, within the source form or documentation, or within a display generated by the derivative works (wherever such third-party notices normally appear). The contents of the NOTICE file do not modify the license, as they are for informational purposes only, and adding more attribution notices as addenda to the NOTICE text is permissible, provided that these notices cannot be understood as modifying the license. Modifications may have appropriate copyright notices, and may provide different license terms for the modifications.

Unless explicitly stated otherwise, any contributions submitted by a licensee to a licensor will be under the terms of the license without any terms and conditions, but this does not preclude any separate agreements with the licensor regarding these contributions.

(E) Which popular software released under this software?

In October 2012, 8,708 projects located at SourceForge.net were available under the terms of the Apache License. In a blog post from May 2008, Google mentioned that over 25% of the nearly 100,000 projects then hosted on Google Code were using the Apache License, including the Android operating system.

(F) Any popular news associated with the license

Wakefield, MA, Aug. 23, 2018 (GLOBE NEWSWIRE) -- The Apache Software Foundation (ASF), the all-volunteer developers, stewards, and incubators of more than 350 Open Source projects and initiatives, announced today Apache® HAWQ® as a Top-Level Project (TLP).

Apache HAWQ is an advanced enterprise SQL-on-Hadoop query engine and analytic database. It combines the key technological advantages of MPP database with the scalability and convenience of Apache Hadoop. HAWQ reads data from and writes data to HDFS natively,

delivers industry-leading performance and linear scalability, and provides users with a complete, standards compliant SQL interface.

Apache HAWQ is in use at Alibaba, Haier, VMware, ZTESoft, and hundreds of users around the world.

(G) Popularity

In October 2012, 8,708 projects located at SourceForge.net were available under the terms of the Apache License. In a blog post from May 2008, Google mentioned that over 25% of the nearly 100,000 projects then hosted on Google Code were using the Apache License, including the Android operating system.

As of 2015, according to Black Duck Software and GitHub, the Apache license is the third most popular license in the FOSS domain after MIT license and GPLv2.

The OpenBSD project does not consider the Apache License 2.0 to be an acceptable license due to its patent provisions.

(H) Impact

The Apache Software Foundation has been home to numerous important open source software projects from its inception in 1999. Successes range from Geronimo to Tomcat to Hadoop, the distributed computing system that now serves as a linchpin of the big data realm.

While Apache does not maintain comprehensive statistics on downloads, the Apache HTTP Server, for example, powers nearly 500 million websites, and OpenOffice, which came into Apache's hands only recently, has been downloaded millions of times. Apache also offers one of the more popular permissive open source licenses.

There are 15 Apache projects that have been critical over the years, not only to the open source movement but to the technology world at large.

GNU (GPL)

(A) History of the License

The GPL was written by Richard Stallman in 1989, for use with programs released as part of the GNU project. The original GPL was based on a unification of similar licenses used for early versions of GNU Emacs (1985), the GNU Debugger and the GNU C Compiler. licenses contained similar provisions to the modern GPL, but were specific to each program, rendering them incompatible, despite being the same license. Stallman's goal was to produce one license that could be used for any project, thus making it possible for many projects to share code.

The second version of the license, version 2, was released in 1991. Over the following 15 years, members of the free software community became concerned over problems in the GPLv2

license that could let someone exploit GPL-licensed software in ways contrary to the license's intent. These problems included tivoization (the inclusion of GPL-licensed software in hardware that refuses to run modified versions of its software), compatibility issues similar to those of the Affero General Public License—and patent deals between Microsoft and distributors of free and open source software, which some viewed as an attempt to use patents as a weapon against the free software community.

Version 3 was developed to attempt to address these concerns and was officially released on 29 June 2007.

(B) Idea of the License

The GNU General Public License, often shortened to GNU GPL (or simply GPL), lists terms and conditions for copying, modifying and distributing free software. The GPL was created by Richard Stallman in order to protect GNU software from being made proprietary. It is a specific implementation of his "copyleft" concept. According to Stallman, copyleft is a derivative of copyright law that serves "the opposite of its usual purpose: instead of a means of privatizing software, it becomes a means of keeping software free."

(C) What problem does it solve?

Its aim is to give computer users freedom and control in their use of their computers and computing devices, by collaboratively developing and providing software that is based on the following freedom rights: users are free to run the software, share it (copy, distribute), study it and modify it. GNU software guarantees these freedom-rights legally (via its license), and is therefore free software; the use of the word "free" always being taken to refer to freedom.

In order to ensure that the entire software of a computer grants its users all freedom rights (use, share, study, modify), even the most fundamental and important part, the operating system (including all its numerous utility programs), needed to be free software. According to its manifesto, the founding goal of the project was to build a free operating system and, if possible, "everything useful that normally comes with a Unix system so that one could get along without any software that is not free." Stallman decided to call this operating system GNU (a recursive acronym meaning "GNU's not Unix"), basing its design on that of Unix, a proprietary operating system. Development was initiated in January 1984. In 1991, the Linux kernel appeared, developed outside the GNU project by Linus Torvalds, and in December 1992 it was made available under version 2 of the GNU General Public License. Combined with the operating system utilities already developed by the GNU project, it allowed for the first operating system that was free software, known as Linux.
The project's current work includes software development, awareness building, political campaigning and sharing of the new material.

(D) Detailed Licensing Model

The GNU Free System Distribution Guidelines (GNU FSDG) is a system distribution commitment used to explain what it means for an installable system distribution (such as a Linux distribution) to qualify as free (libre), and help distribution developers make their distributions qualify.

Mostly this includes distributions that are a combination of GNU packages with a Linux-libre kernel (a modified Linux kernel, that removes binary blobs, obfuscated code and portions of code under proprietary licenses) and consist only of free software (eschewing proprietary software entirely). Distributions that have adopted the GNU FSDG includes Dragora GNU/Linux-libre, gNewSense, Parabola GNU/Linux-libre, Trisquel GNU/Linux, Ututo, and a few others.

The Fedora Project distribution license guidelines were used as a basis for the FSDG.

(E) Popular Software Releases

GnuPG 2.2.10 released, Werner Koch, 10:43

GNU AutoGen Version 5.18.16/AutoOpts Version 42.1, Bruce Korb, 09:14

dico-2.6 released [stable], Sergey Poznyakoff, 09:21

GNU Octave 4.4.1 Released, John W. Eaton, 16:46

GCC 8.2 Released, Jakub Jelinek, 08:42

(F) Popular news

Following five years of hectoring, Tesla has released a portion of the open-source code it's obligated to provide under the terms of the GNU General Public License (GPL).

Since 2013, the Software Freedom Conservancy (SFC), responding to complaints of GPL violations related to software in the Tesla Model S, has pressed the carmaker to comply with the terms of the GPL.

The SFC provides legal support to open source projects. In theory, Tesla could be sued for flouting the GPL, but even the SFC, which backed the controversial GPL claim against VMware, prefers resolving compliance issues outside of court.

(G) Popularity

Historically, the GPL license family has been one of the most popular software licenses in the FOSS domain.

A 1997 survey of MetaLab, then the largest free software archive, showed that the GPL accounted for about half of the software licensed therein. Similarly, a 2000 survey of Red Hat Linux 7.1 found that 53% of the source code was licensed under the GPL. As of 2003, about

68% of all projects and 82.1% of the open source industry certified licensed projects listed on SourceForge.net were from the GPL license family. As of August 2008, the GPL family accounted for 70.9% of the 44,927 free software projects listed on Freecode.

GPL usage statistics from 2009 to 2013 was extracted from Freecode data by Walter

van Holst
while analyzing license proliferation.

(H) Impact

So what effect has the GPL had on software development and the open source movement as a whole? It's actually had a tremendous impact.

First, many very common UNIX applications, such as GNU Emacs, have been released

under
the GPL, and are used by countless numbers of users every day.
Second, the open source software movement has taken several ideas promoted by the

GPL and
modified them slightly. The most important is the idea that software licensing should include
access to source code.

Mozilla Public License

(A) History

Version 1.0 of the MPL was written by Mitchell Baker in 1998 while working as a lawyer at Netscape Communications Corporation. Netscape was hoping an open source strategy for developing their own Netscape web browser would allow them to compete better with Microsoft's browser, Internet Explorer. To cover the browser's code, the company drafted a license known as the Netscape Public License (NPL), which included a clause allowing even openly developed code to be theoretically relicensed as proprietary.

However, at the same time, Baker developed a second license similar to the NPL. It was called the Mozilla Public License after Netscape's project name for the new open source codebase, and although it was originally only intended for software that supplemented core modules covered by the NPL, it would become much more popular than the NPL and eventually earn approval from the Open Source Initiative.

Less than a year later, Baker and the Mozilla Organization would make some changes to the MPL, resulting in version 1.1, a minor update.

(B) Idea of the license

The Mozilla Public License (MPL) is a free and open source software license developed and maintained by the Mozilla Foundation. It is a weak copyleft license, characterized as a middle ground between permissive free software licenses and the GNU General Public License (GPL), that seeks to balance the concerns of proprietary and open source developers.

It has undergone two revisions, a minor update to version 1.1, and a major update to version 2.0 with the goals of greater simplicity and better compatibility with other licenses.

(C) What problem does it solve

The MPL fills a useful space in the spectrum of free and open source software licenses, sitting between the Apache license, which does not require modifications to be shared, and the GNU family of licenses, which requires modifications to be shared under a much broader set of circumstances than the MPL.

(D) Detailed licensing model

The MPL defines rights as passing from "Contributors" who create or modify source code, through an optional auxiliary distributor (themselves a licensee), to the licensee. It grants liberal copyright and patent licenses allowing for free use, modification, distribution, and "exploit[ation]" of the work, but does not grant the licensee any rights to a contributor's trademarks. These rights will terminate if the licensee fails to comply with the license's terms and conditions, but a violating licensee who returns to compliance regains their rights, and even receiving written notice from a Contributor will result in losing rights to that Contributor's code only. A patent retaliation clause, similar to that of the Apache License, is included to protect an auxiliary distributor's further recipients against patent trolling. The contributors disclaim warranty and liability, but allow auxiliary distributors to offer such things on their own behalf.

It is explicitly granted that MPL-covered code may be distributed under the terms of the

license
version under which it was received, or any later version. If code under version 1.0 or
1.1 is
upgraded to version 2.0 by this mechanism, the 1.x-covered code must be marked with

the
aforementioned GPL-incompatible notice. The MPL can be modified to form a new
license,
provided that said license does not refer to Mozilla or Netscape.

(E) Popular software releases

Bugzilla

Chatzilla

Firefox

Firefox focus

Mozilla Thunderbird

Mozilla Sunbird

Waterfox

(F) Popular News

Mozilla has launched a test of a new Firefox add-on that recommends sites based on the user's current and past browsing.

The add-on, dubbed "Advance," was the latest in the Test Pilot project that Mozilla has run, under that name and others, since 2015. "Test Pilot is a way for you to try out experimental features and let us know what you think," said Nick Nguyen, vice president of Firefox, in a May 2016 blog post.

(G) Popularity

The MPL is the license for Mozilla Firefox, Mozilla Thunderbird, and most other Mozilla software,[11] but it has been used by others, such as Adobe to license their Flex product line,[12] and The Document Foundation to license LibreOffice 4.0 (also on LGPL 3+).[13][14] Version 1.1 was adapted by several projects to form derivative licenses like Sun Microsystems' own Common Development and Distribution License.

(H) Impact

The Mozilla Public License (MPL) is a free and open source software license developed and maintained by the Mozilla Foundation. It is a weak copyleft license, characterized as a middle ground between permissive free software licenses and the GNU General Public License (GPL), that seeks to balance the concerns of proprietary and open source developers.

It has undergone two revisions, a minor update to version 1.1, and a major update to version 2.0 with the goals of greater simplicity and better compatibility with other licenses.

Bibliography:

www.google.com

www.wikipedia.org

www.encyclopedia.com

www.zdnet.org

www.quora.com

Practical 3

Contribution to Wikipedia

Introduction

Wikipedia has grown to be a very popular and well-known online resource. All of the content on the site is created and edited by and offered under a Creative Commons Attribution / Sharealike 3.0 license. It has:

- Over 4.6 million articles;
- Over 33.9 million pages; and
- Over 22.7 million user accounts with 73,251 active users as of May 2014.

A 2010 survey found that "[a]bout 23% of contributors have completed degree-level education, 26% are undergraduates and 45% have secondary education or less. 87% are men and 13% women."

Wikipedia is a a good source of:

- Basic reference information on a wide range of topics;
- Links to sources to start your academic research on a topic; and
- Creative Commons-licens

Operating model

An organization is a complex system for delivering value. An operating model breaks this system into components, showing how it works. It can help different participants understand the whole. It can help leaders identify problems that are causing under performance. It can help those making

changes check that they have thought through all elements and that the whole will still work. It can help those transforming an operation coordinate all the different changes that need to happen.

An operating model is like the blueprint for a building. It is more dynamic than a building blueprint, with changes occurring regularly. Also, an operating model is not usually just one blueprint. There are likely to be blueprints for each element: processes, organization, decision making, software applications, locations and so on.

An operating model can describe the way an organization does business today – the *as is*. It can also communicate the vision of how an operation will work in the future – the *to be*. In this context it is often referred to as the target operating model, which is a view of the operating at a future point in time. Most typically, an operating model is a living set of documents that are continually changing, like an organization chart.

An operating model describes how an organization delivers value, as such it is a subset of the larger concept 'business model'. A *business model* describes how an organization creates, delivers and captures value and sustains itself in the process. An operating model focuses on the delivery element of the business model. There are plenty of disagreements about the use of the words *business model* and *operating model*

License

As per the licensing update vote result and subsequent (May 2009) Wikimedia Foundation Board resolution, any content on Wikimedia Foundation projects available under the GNU Free Documentation License 1.2 with the possibility of upgrading to a later version was made available additionally under Creative Commons Attribution-ShareAlike 3.0 Unported License (CC-BY-SA). Specifically with regard to text, since this update, only dual-licensed content or CC-BY-SA-compatible content can be added to the projects, any GFDL-only submissions will no longer be accepted. In other words, CC-BY-SA is the primary Wikimedia license for text, and GFDL is retained as a secondary license.

As per the Board resolution, this licensing change began to be implemented on all projects on June 15, 2009. (Any content not updated by this date could still be updated on any date prior to August 1, 2009.)

How to Contribute

Create a Wikipedia account. Account creation is not required; however, if you register for an account, you will be given more privileges than a non-registered user. For all of these privileges to take place, your account must be at least four days old and have at least ten edits.

Expand stubs. An article that is not complete, or written in full detail, may be marked with a {{Stub}} tag. You can help by adding content to the articles currently marked as stubs. Articles may also have a more detailed {{stub}} meaning that the stub has been sub-sorted. Sub-sorted stubs include anything from Arts, Cultures, Design, Broadcast Media, Radio, Television, Literature, and more!

Add a photo. An encyclopedia is not complete without pictures. You can upload as many pictures as you wish; however, you must provide detailed information on the source and the license of the file. If you cannot provide that information, do not upload any photos. If you still choose to upload photos, they will be deleted.

Add a photo. An encyclopedia is not complete without pictures. You can upload as many pictures as you wish; however, you must provide detailed information on the source and the license of the file. If you cannot provide that information, do not upload any photos. If you still choose to upload photos, they will be deleted.

Write a new article. Wikipedia is rapidly growing, although the growth has slowed down since 2008. The English version of Wikipedia currently has over 5,168,000 articles! You can help continue this growth by writing an article of your own. You should write an article about something you are very knowledgeable about, so that you can write a complete and informative article. Articles created as test pages, pure vandalism, attack pages, etc. will be deleted on the spot, without any further debate.

Remove spam. Wikipedia is accessed by millions of people every day, thus, there tends to be a lot of vandalism or spamming. People who vandalize or spam a page may have added inappropriate links, blanked the page, added nonsense, etc. You can help out by removing, or reverting, this vandalism. Removing vandalism will make Wikipedia a better place for people to gather information and resources.

Help out. Even though Wikipedia is an encyclopedia, it is also a community. You can help out newcomers to make it a bigger and better community for the encyclopedia.

Do some maintenance. You can help Wikipedia run optimally by doing maintenance tasks like removing copyright violations, fixing up articles, participating in deletion processes, and all sorts of other things.

Revert vandalism. Use tools if you like. If you're getting good at it you'll get a tool called 'rollback' that lets you revert vandalism faster. Remember, spam is also vandalism! If somebody is persistent about vandalising a page, report them to the 'Administrator Intervention against Vandalism' board - AIV for short - once they have been warned suitably.

Practical No. 4: Github

The **Hello World** project is a time-honoured tradition in computer programming. It is a simple exercise that gets you started when learning something new. Let's get started with GitHub!

We will learn how to:

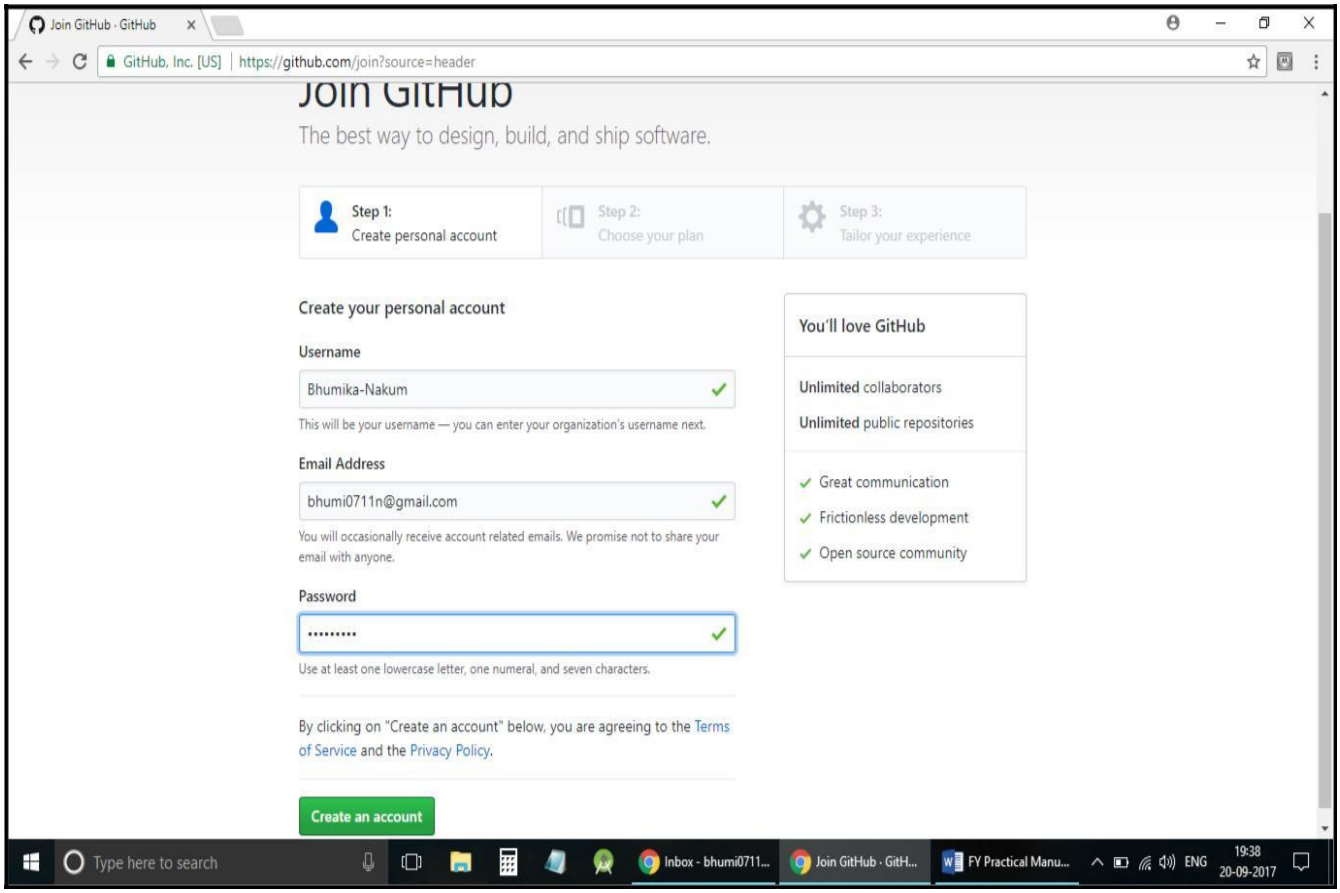
Create and use a repository
Start and manage a new branch
Make changes to a file and push them to GitHub as commits
Open and merge a pull request

What is GitHub?

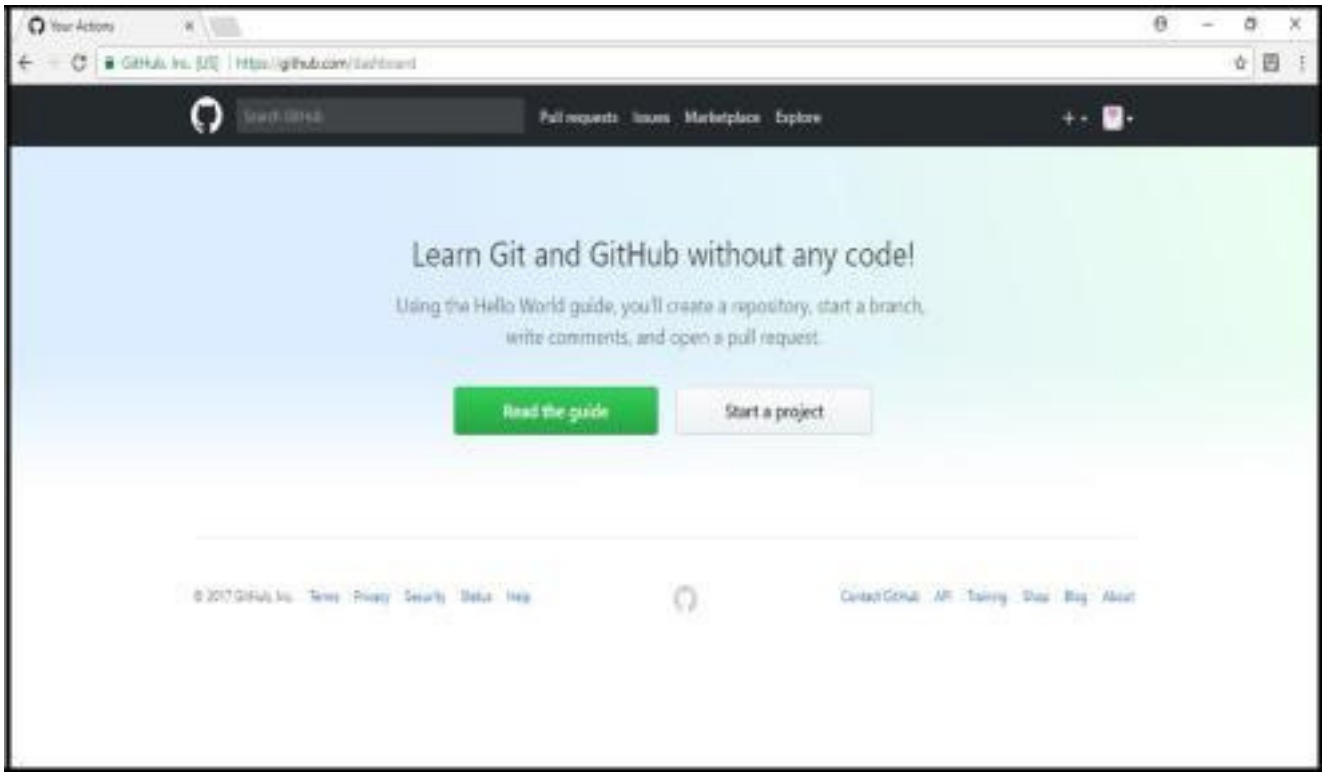
GitHub is a code hosting platform for version control and collaboration. It lets us and others work together on projects from anywhere.

Let us follow the below steps one by one.

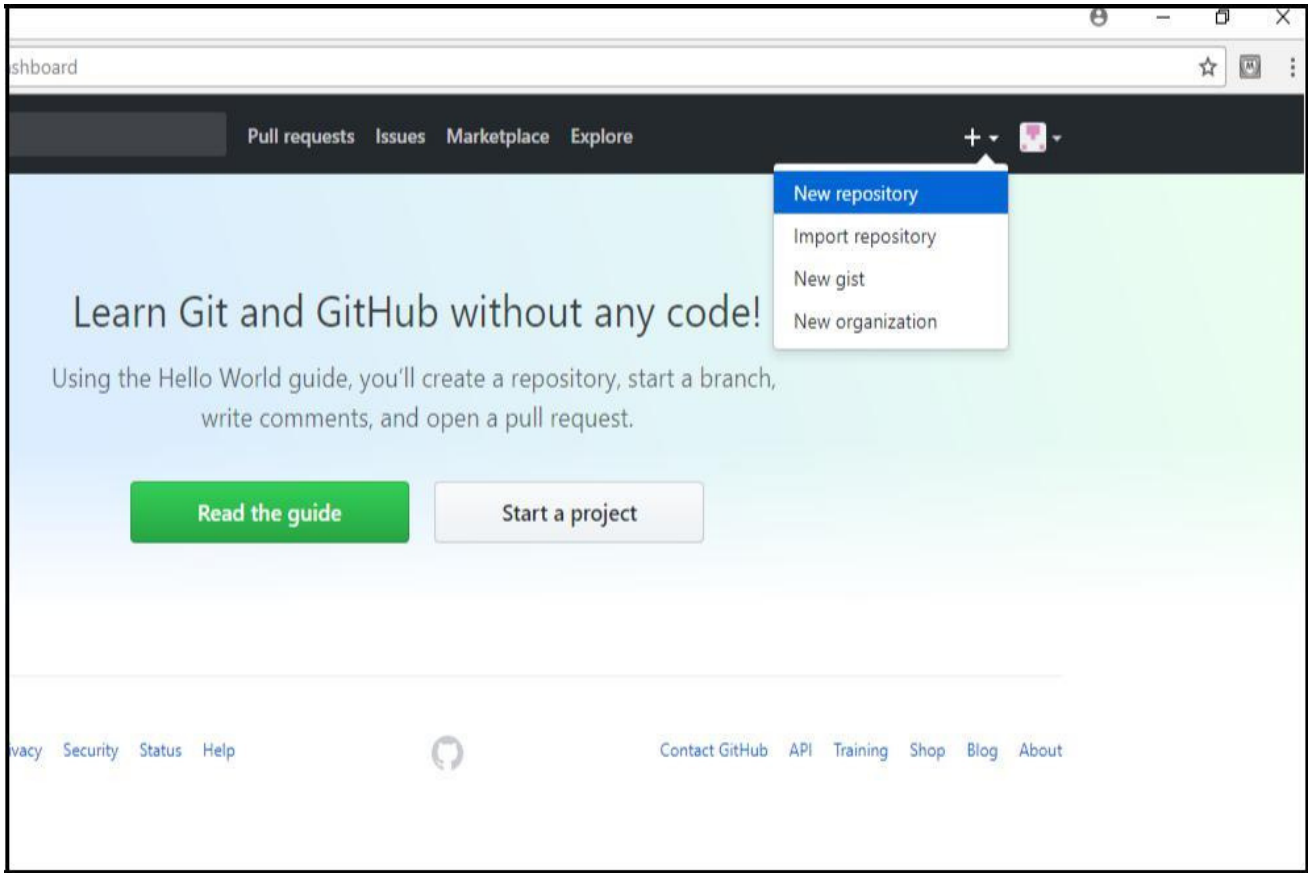
Step1: Open www.github.com and create your account in Github.



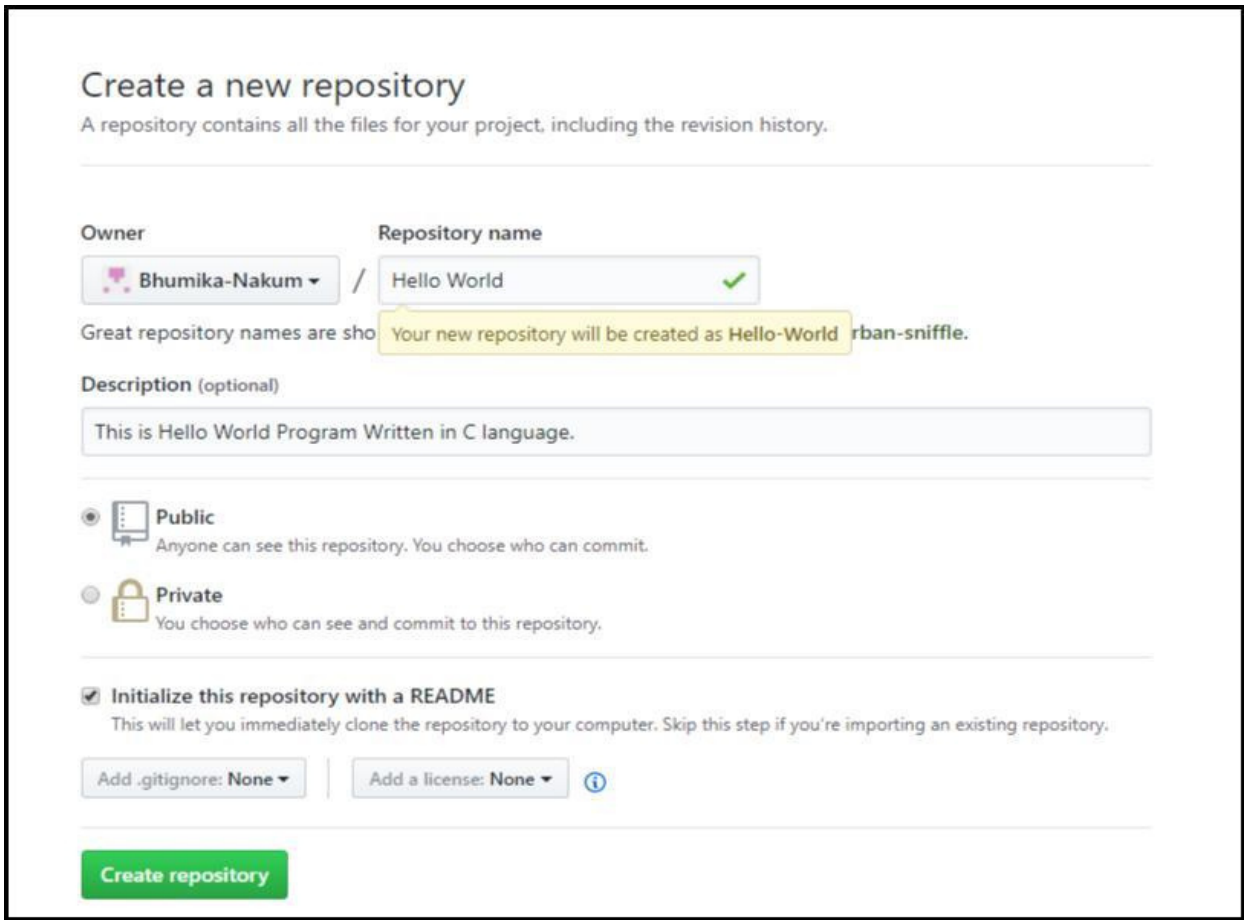
Step2: Enter your ID and Password and Login to your account and get your account verified.



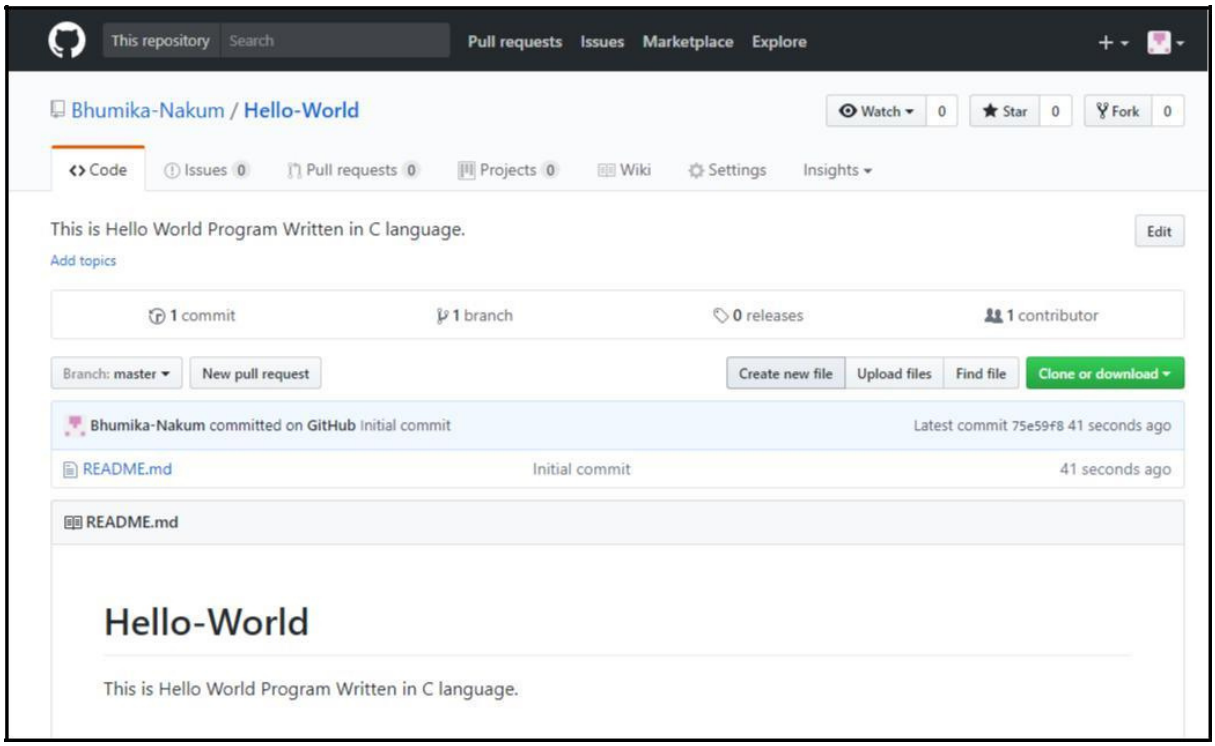
Step 3: Create a new Repository by clicking on '+' sign near your avatar.



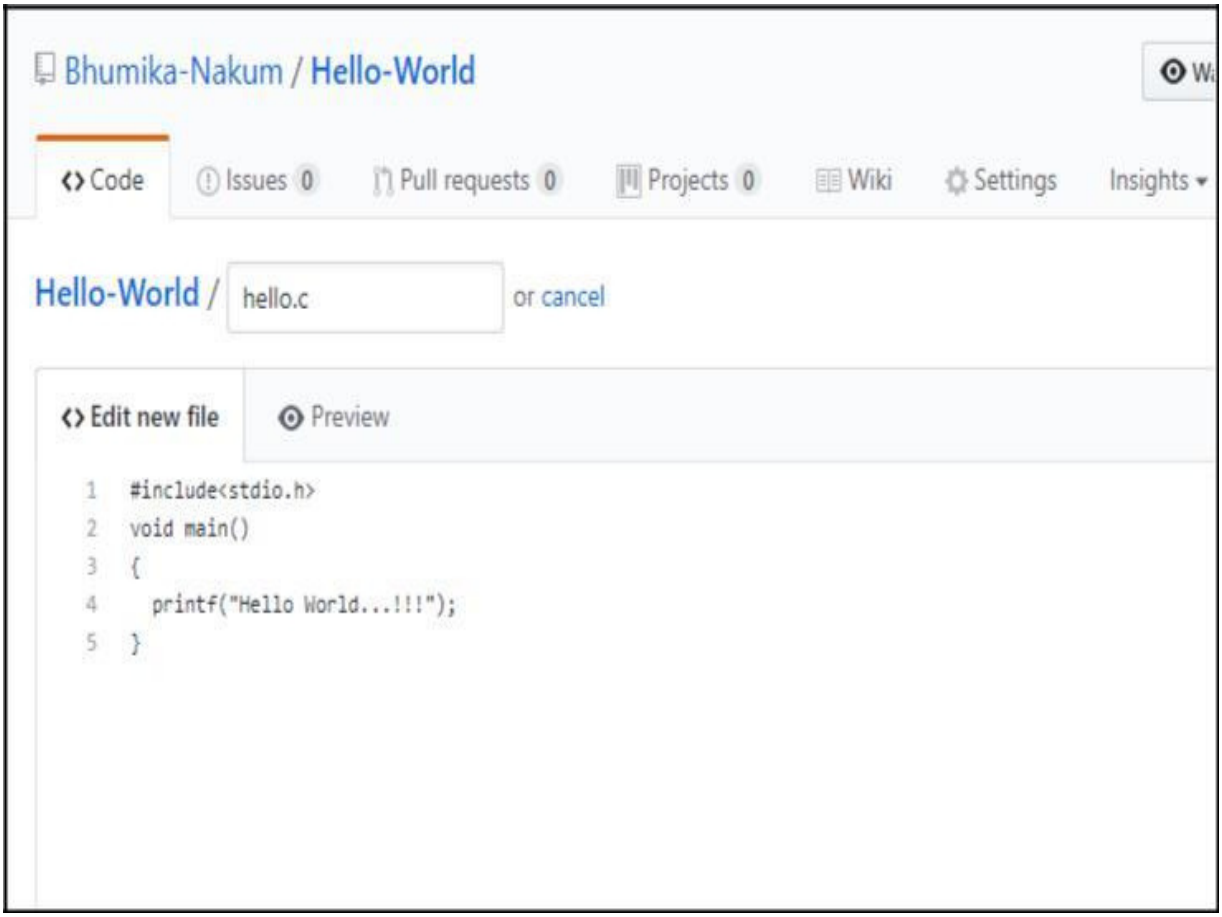
Step 4: Give the name for your repository and initialize the repository with a README. Click on “Create repository”.



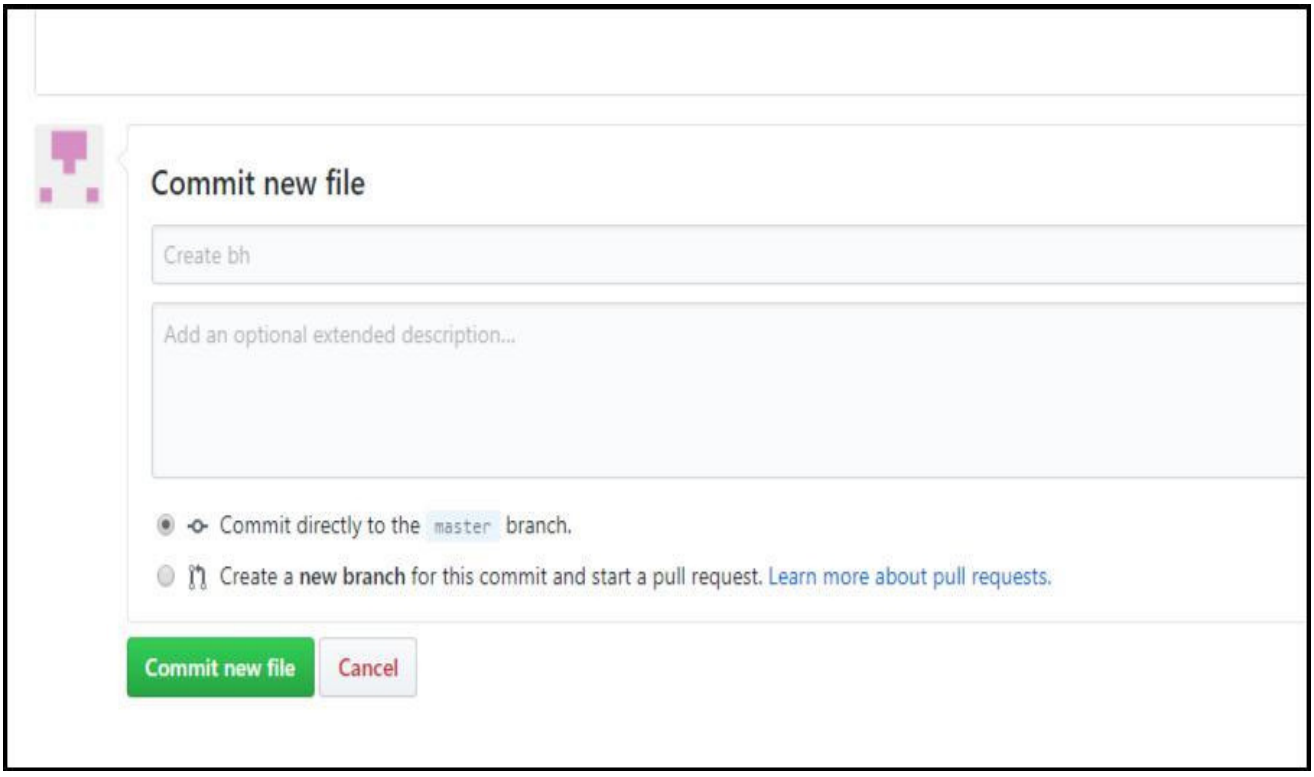
Step 5: Your repository will be created and Master branch will be displayed which contains a file named README.md. This file contains the description of your repository. You can create a new file by clicking on “Create new file” button.



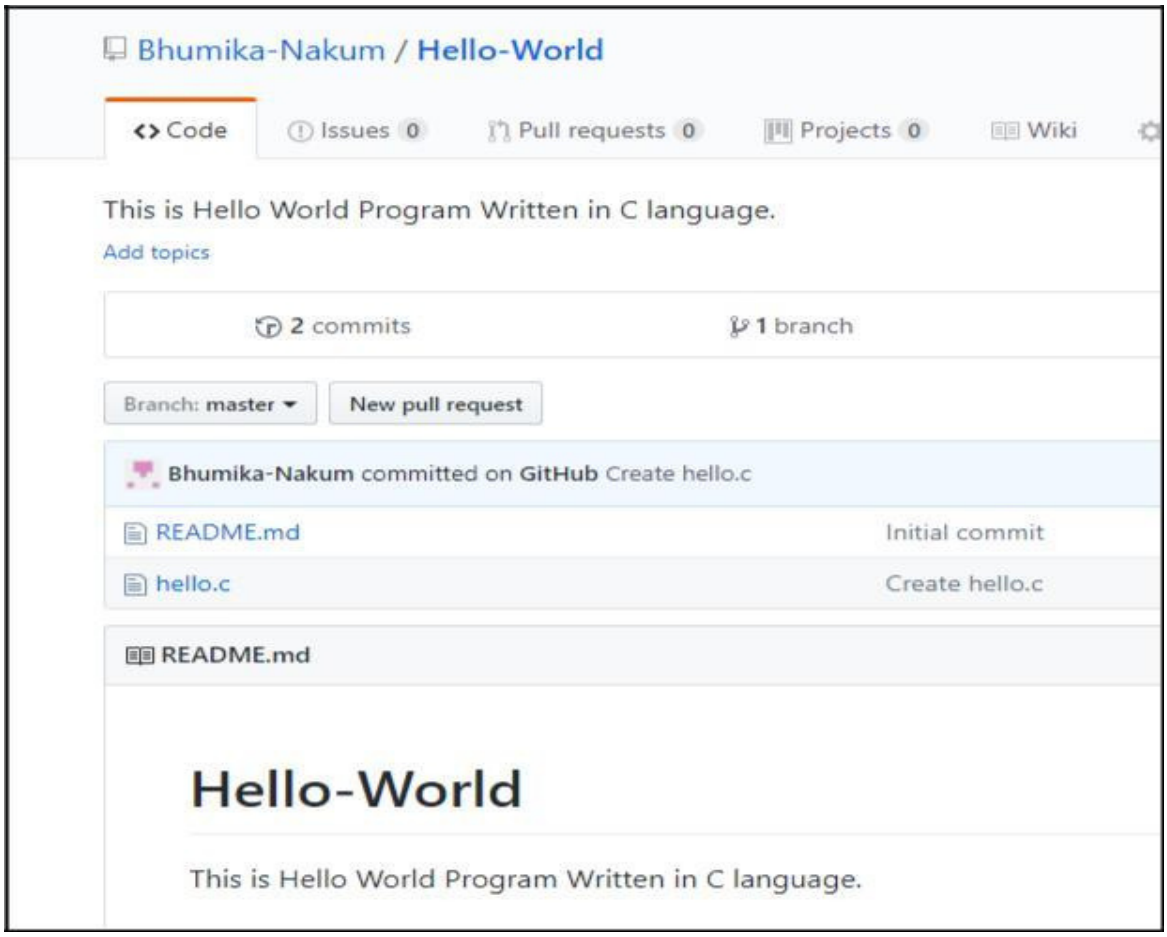
Step 6: Let us create a new file. We will write a basic Hello World program in C language. Give the name of the file as hello.c and write the code in that file.



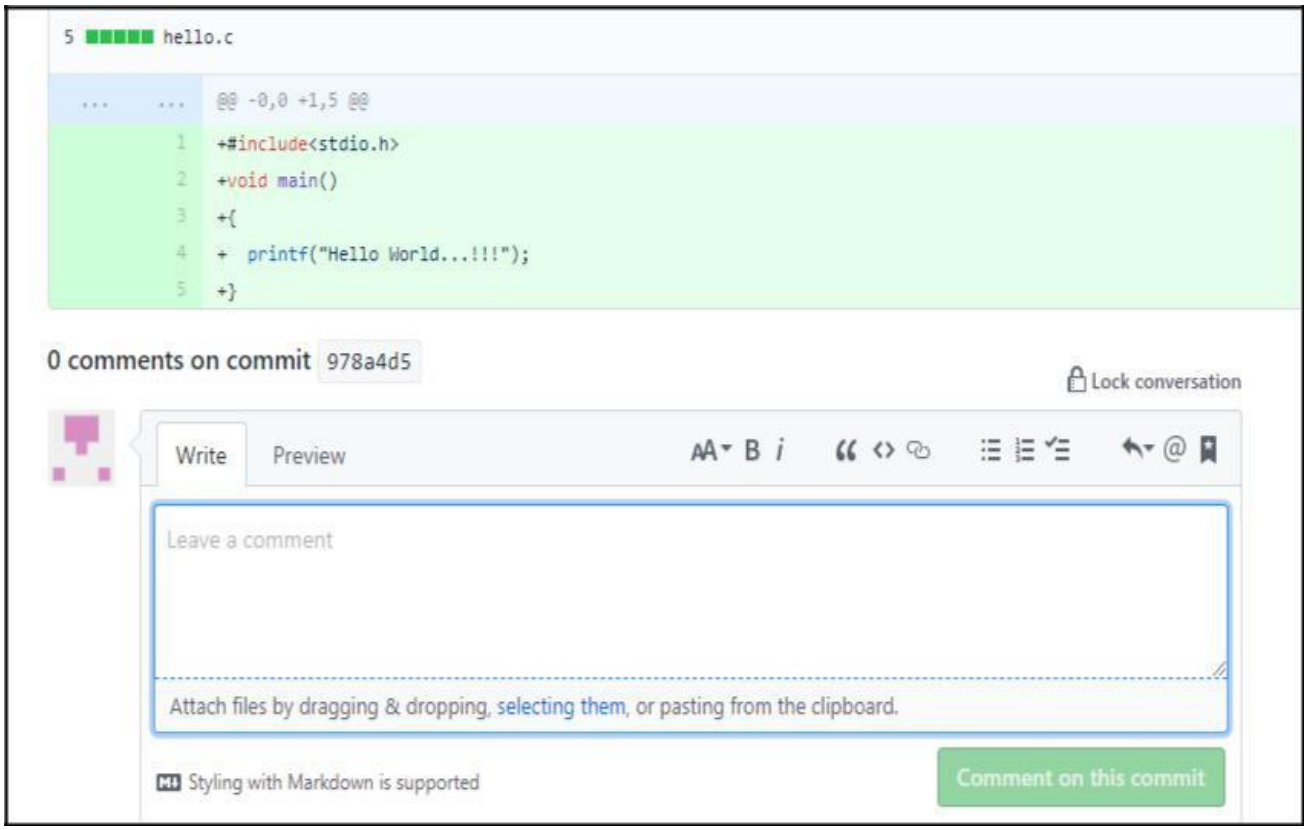
Step 7: After writing the code save the file by clicking on “Commit new file” button which appears in the bottom of that page.



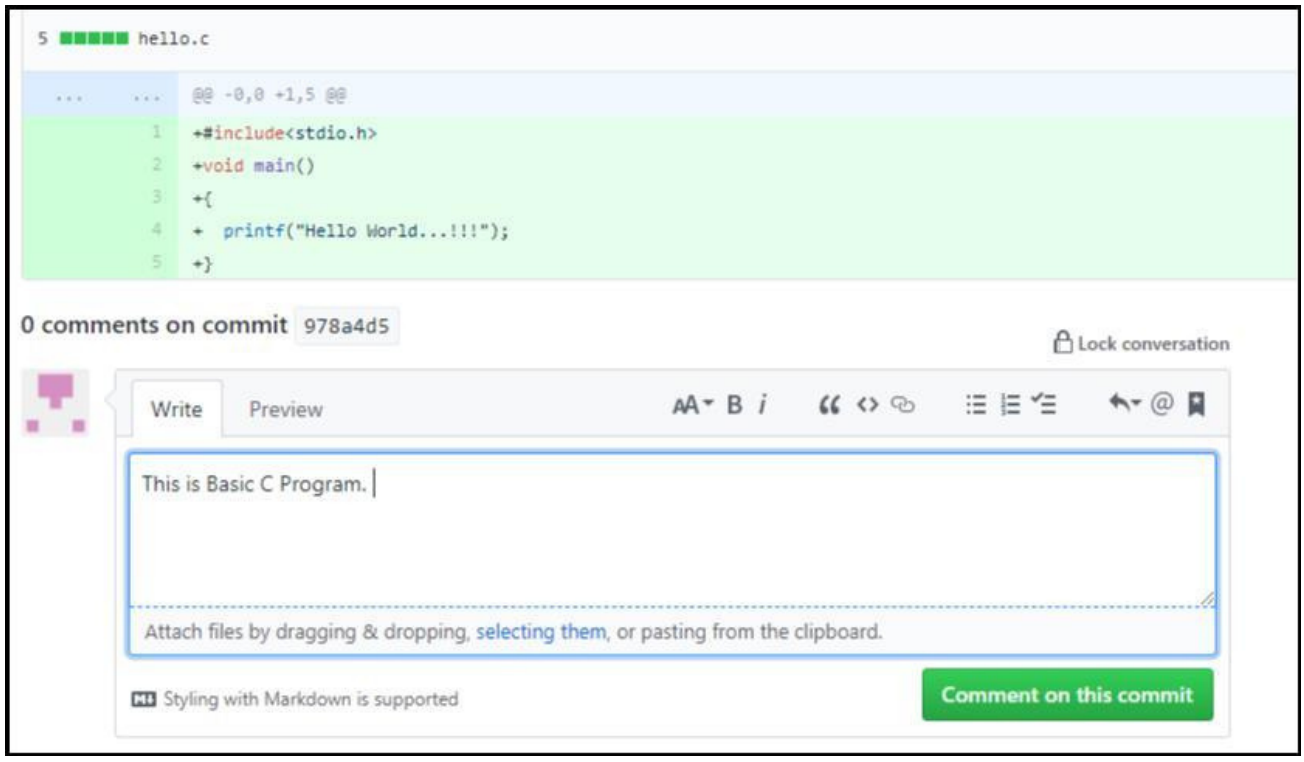
Step 8: You can see a new file has been created with the name “hello.c” in the Master branch. You can add comment on this file by clicking on “Create hello.c”



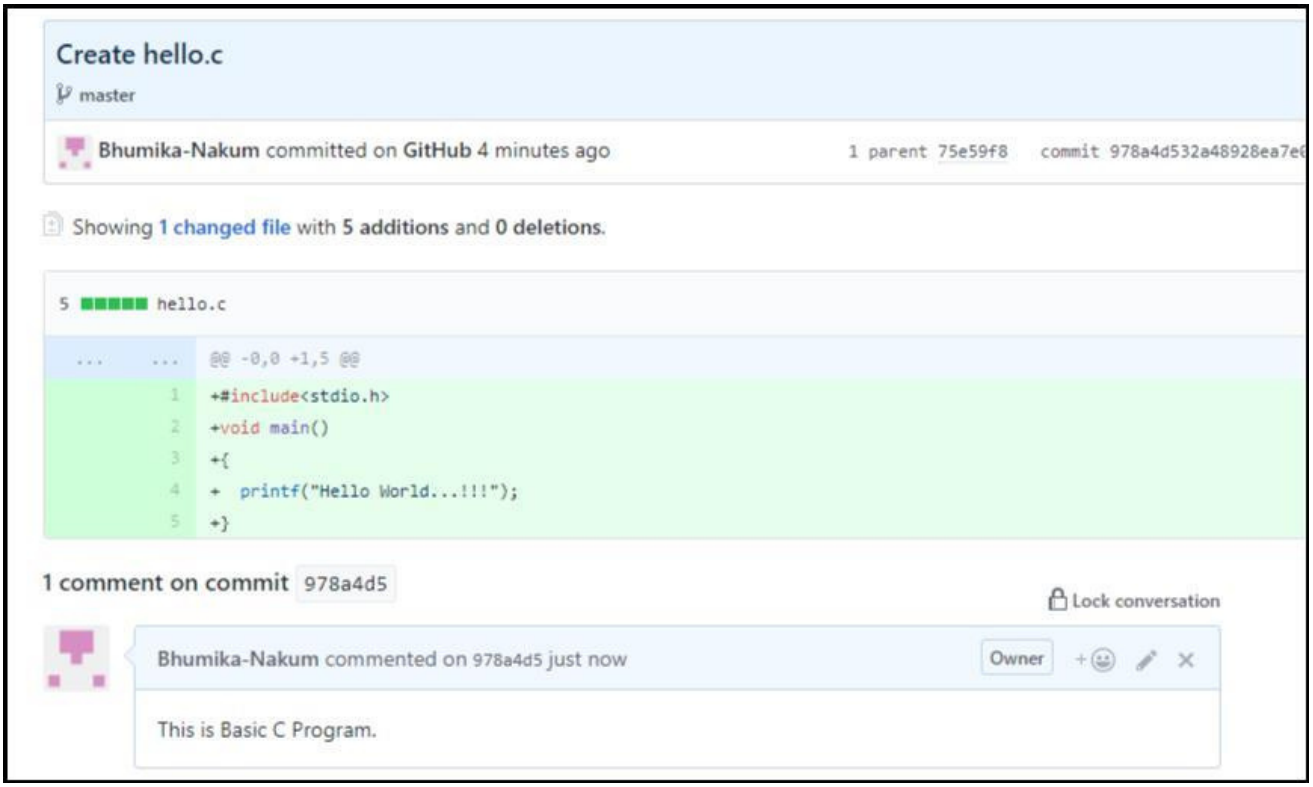
Step 9: After clicking on “Create hello.c” comment section will be displayed.



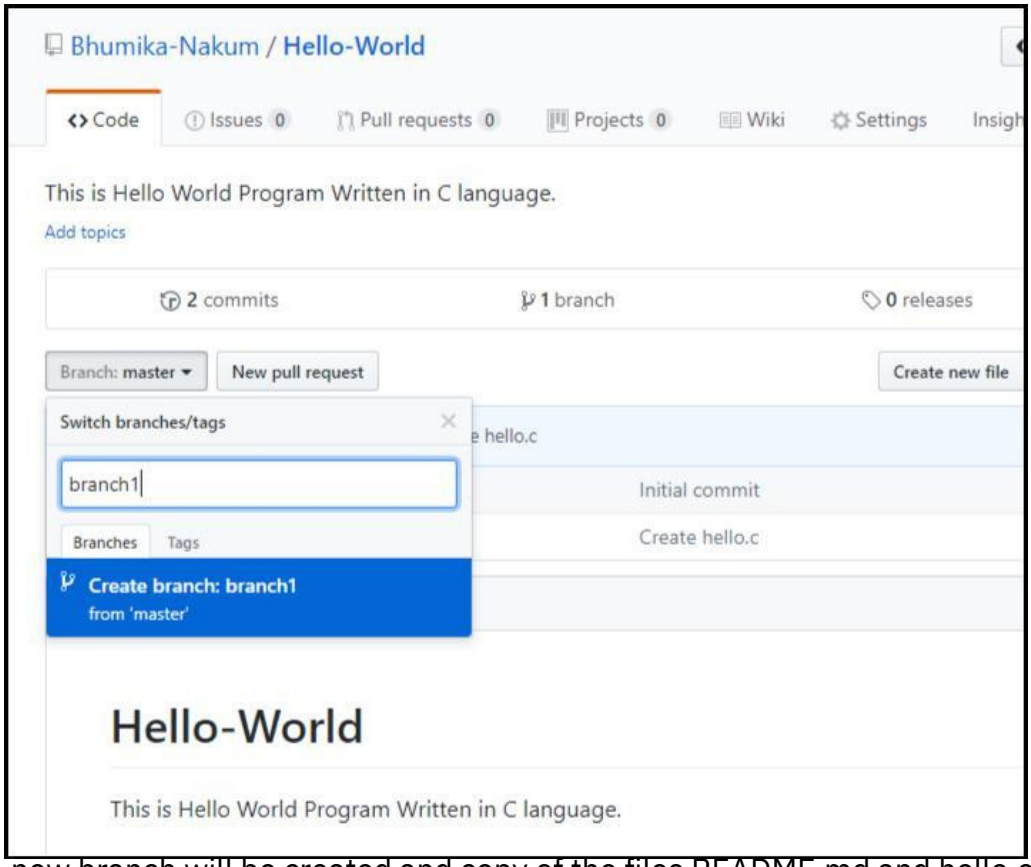
Step 10: You can write the comments related to the file. After writing the comment click on “Comment on this commit” button.



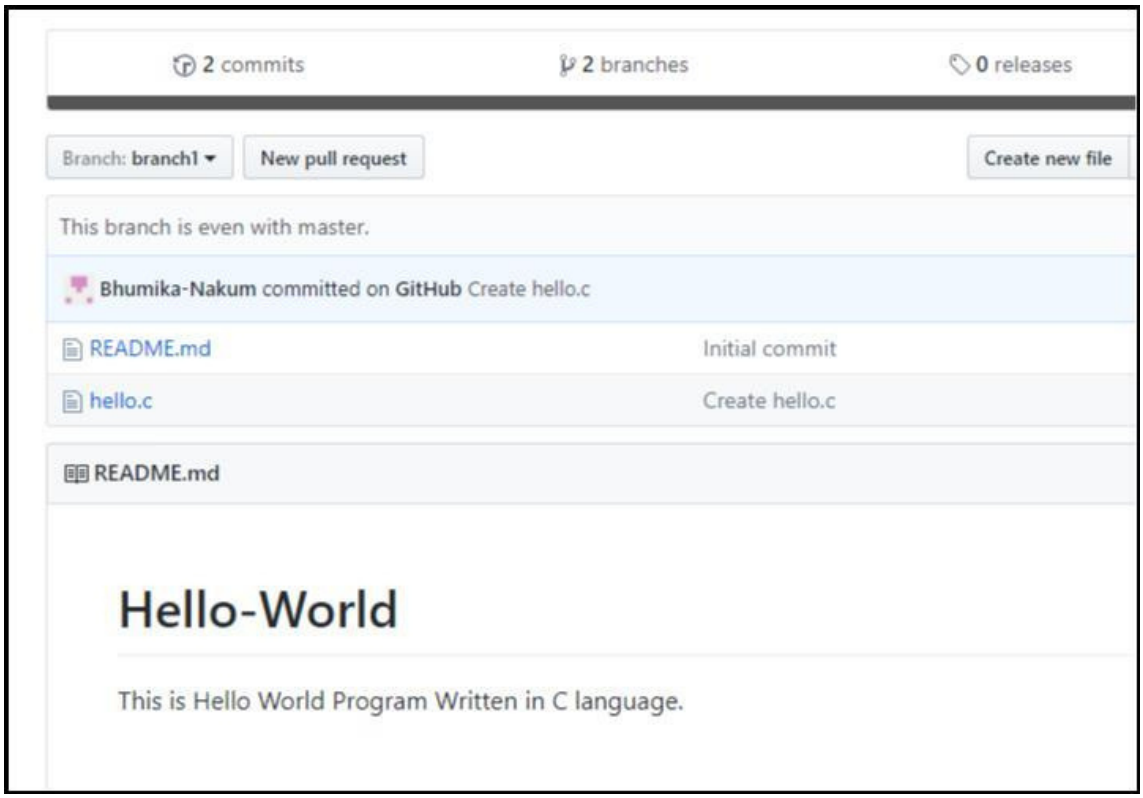
Step 11: The comment will be displayed successfully. Now go back to main page by clicking on the name of your repository (Hello-World) which is displayed on the top of the page.



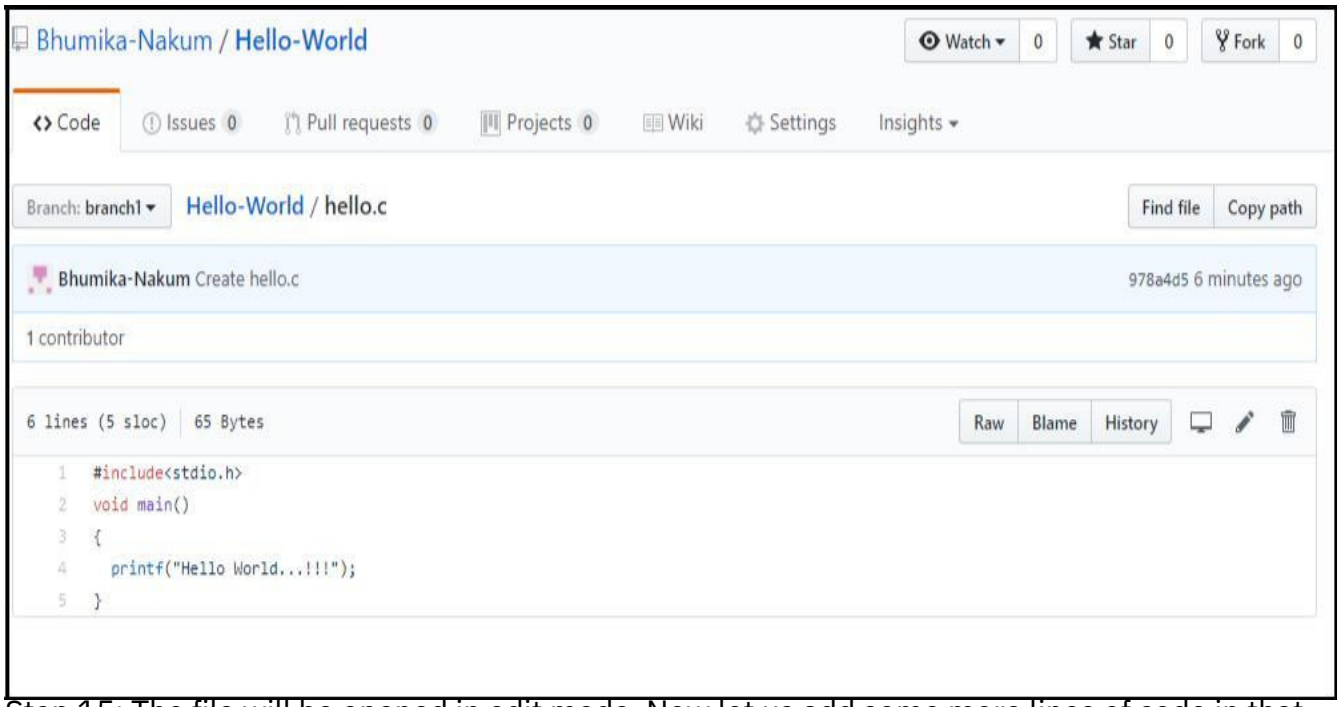
Step 12: Now we will create a new branch. Click on Branch and a drop-down list will appear. Write the name of the Branch as branch1 and click on “Create branch: branch1”.



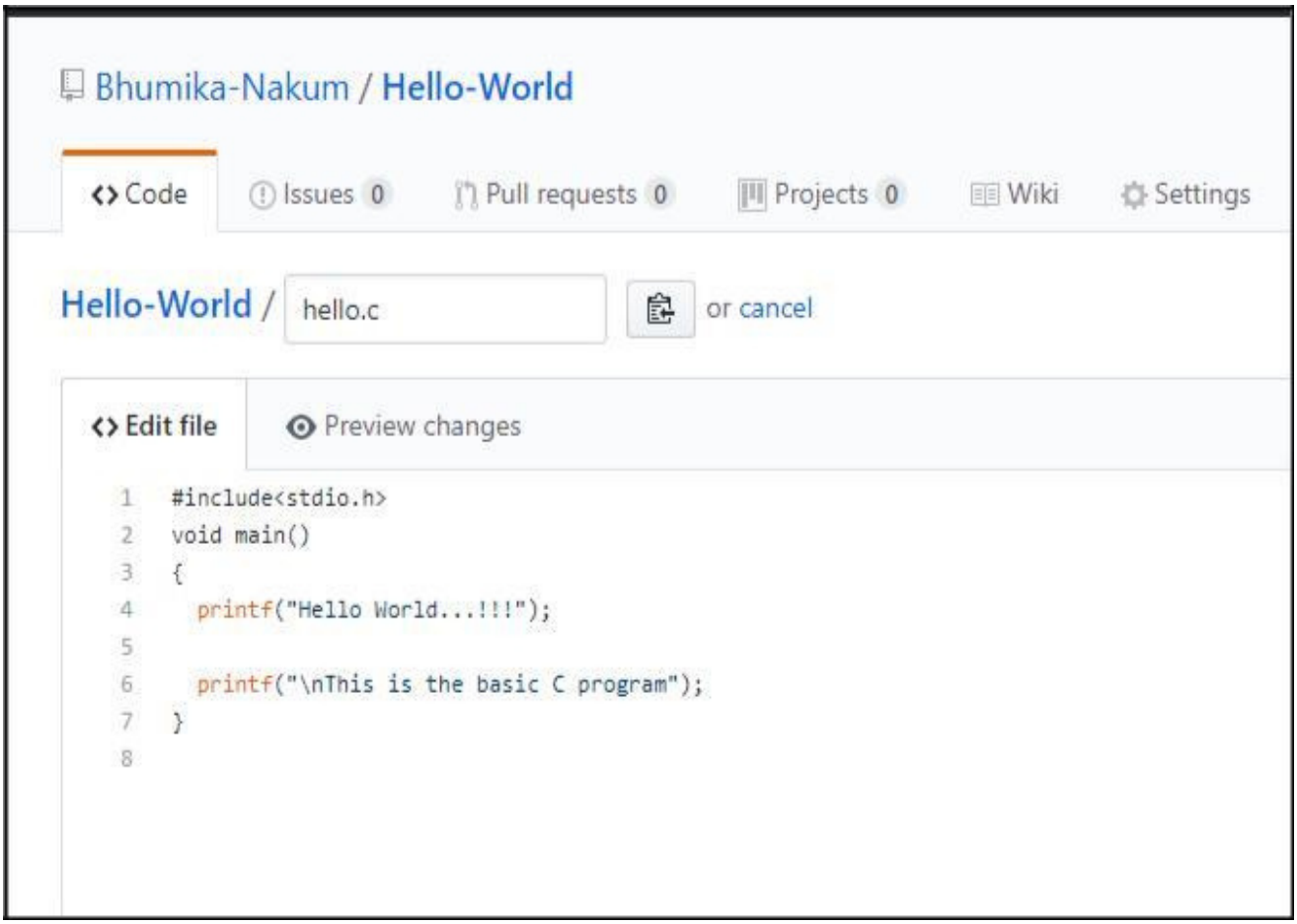
Step 13: A new branch will be created and copy of the files README.md and hello.c will be included in branch1. Now let us edit the file which is in branch1. Click on the file “hello.c”



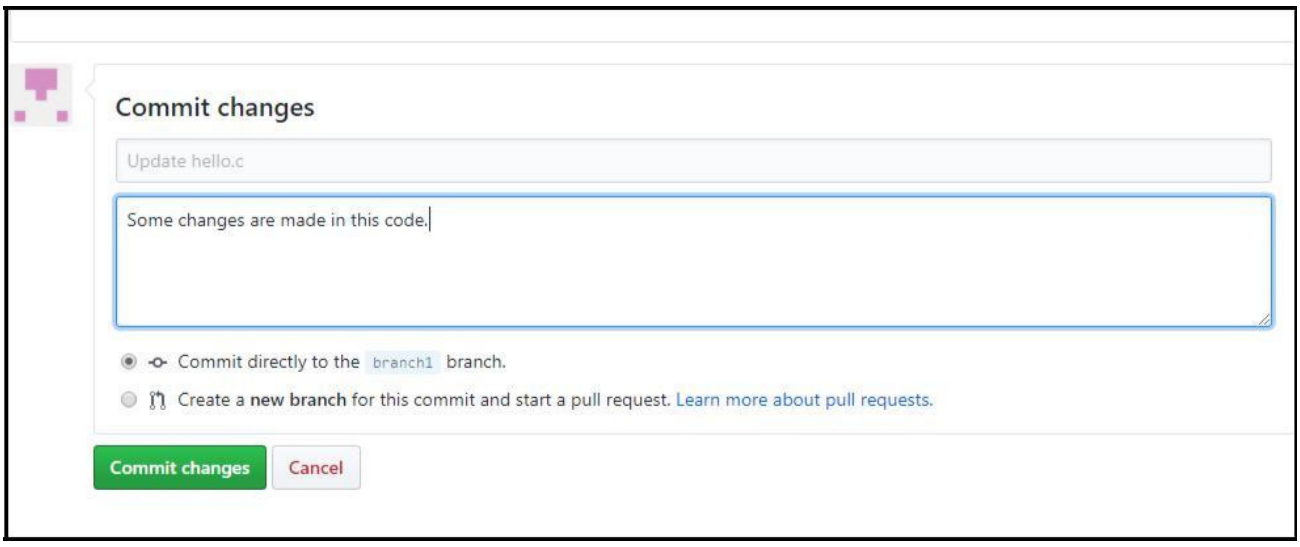
Step 14: The file will be opened. Now click on the edit button (pencil icon) to make the changes in the file.



Step 15: The file will be opened in edit mode. Now let us add some more lines of code in that file.

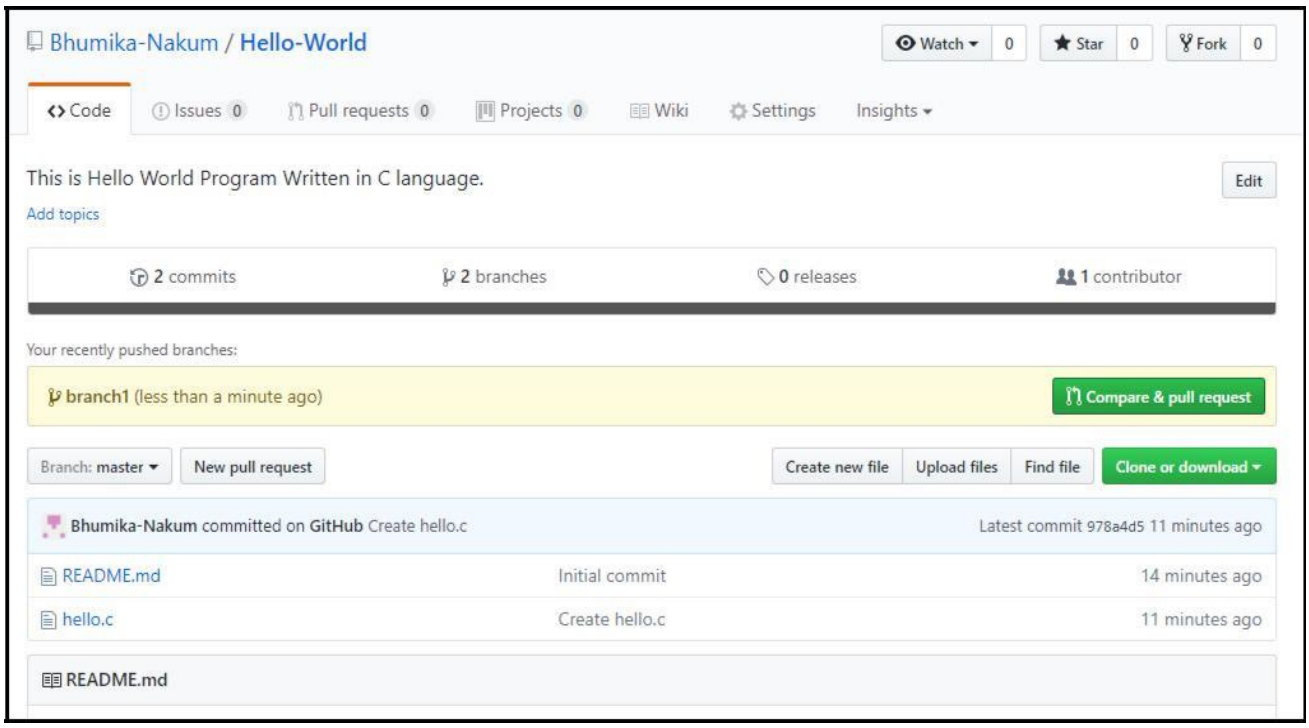


Step 16: After making the changes, click on “Commit changes” button (which in the bottom) to save the changes you have made. The changes will be saved successfully.

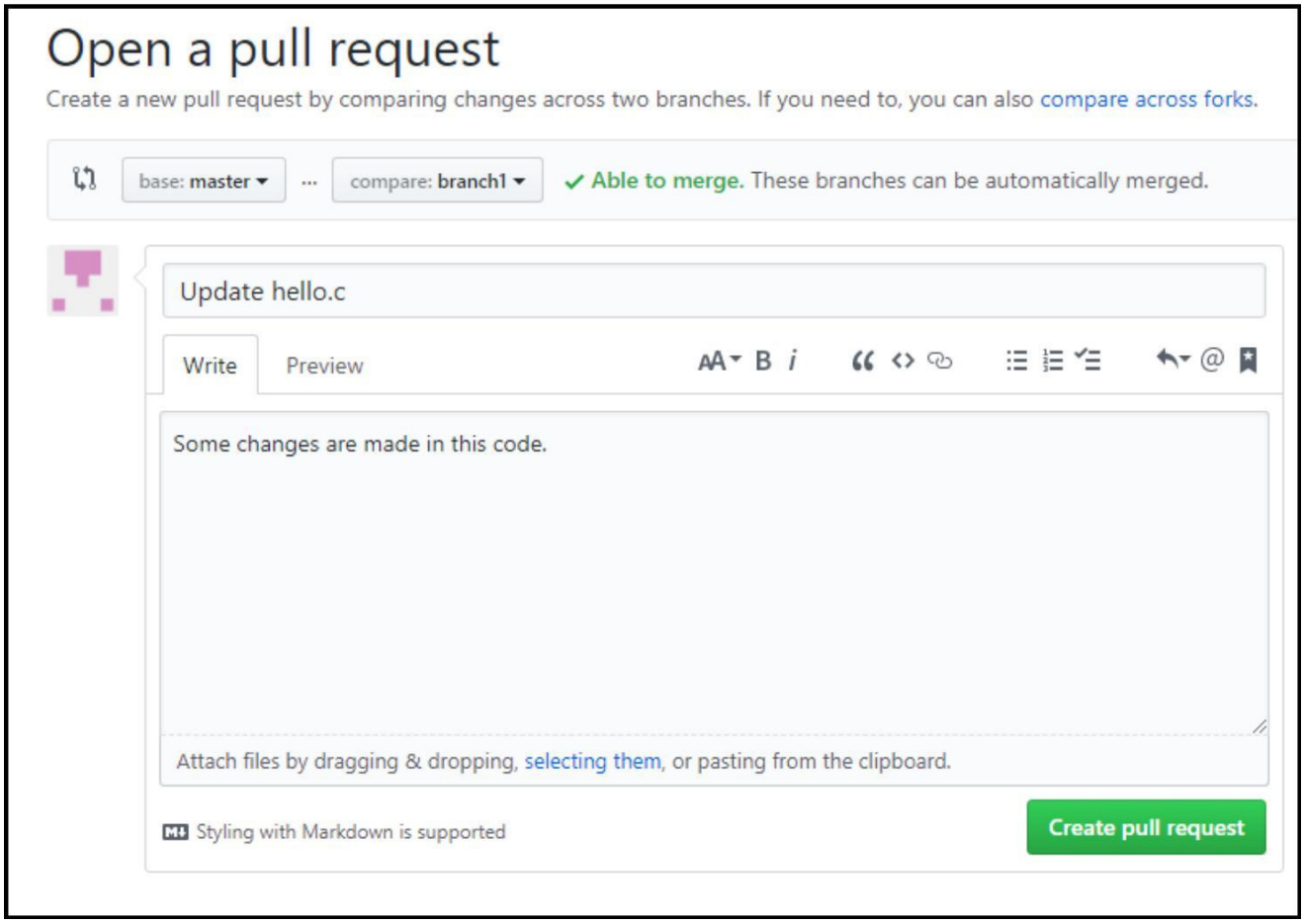


Let us create Pull Request now. Pull requests are proposed changes to a repository submitted by a user and accepted or rejected by a repository's collaborators.
Let us create the Pull Request for branch1 which is created recently. By creating Pull Request for branch1 you are accepting the changes that have made in the file “hello.c”. If the changes are valid, Pull Request will be created successfully and then you can Merge the changes in your original file by creating Merge Request.

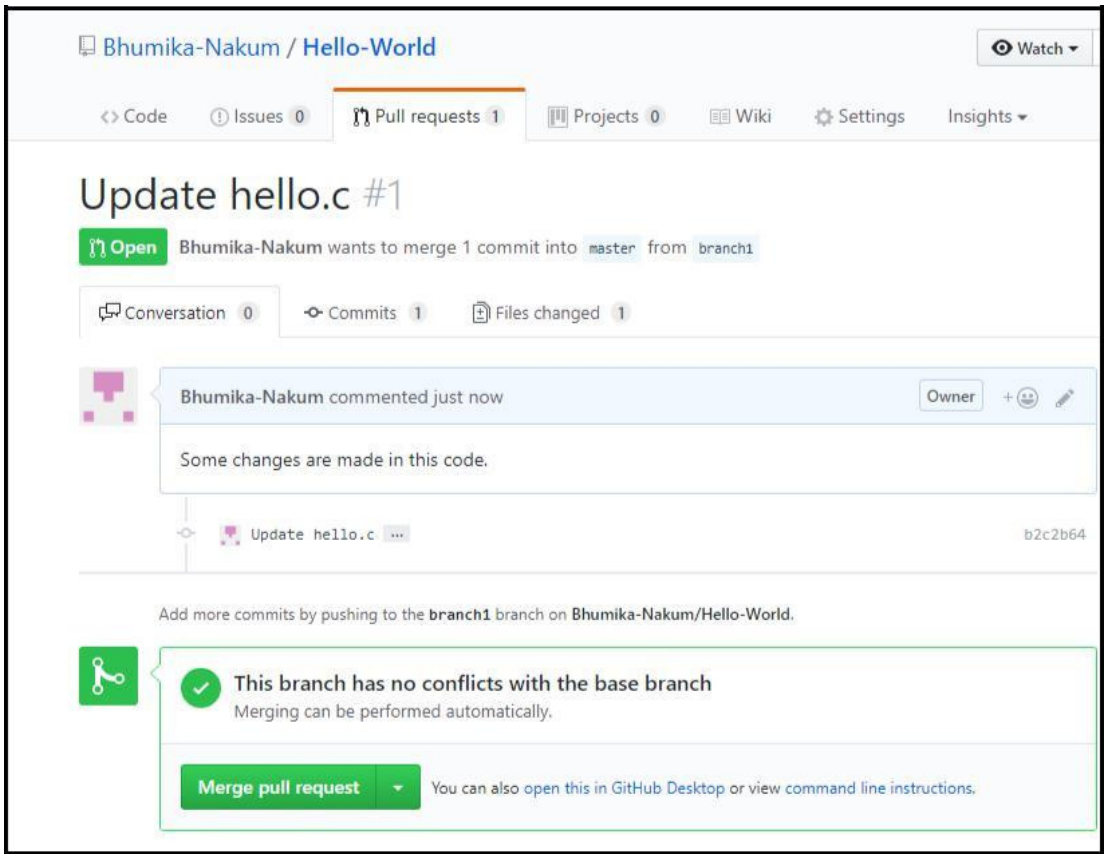
Step 17: Click on the Repository name (Hello-World) which is on the top of the page.
Click on “Compare & pull request” for branch1.



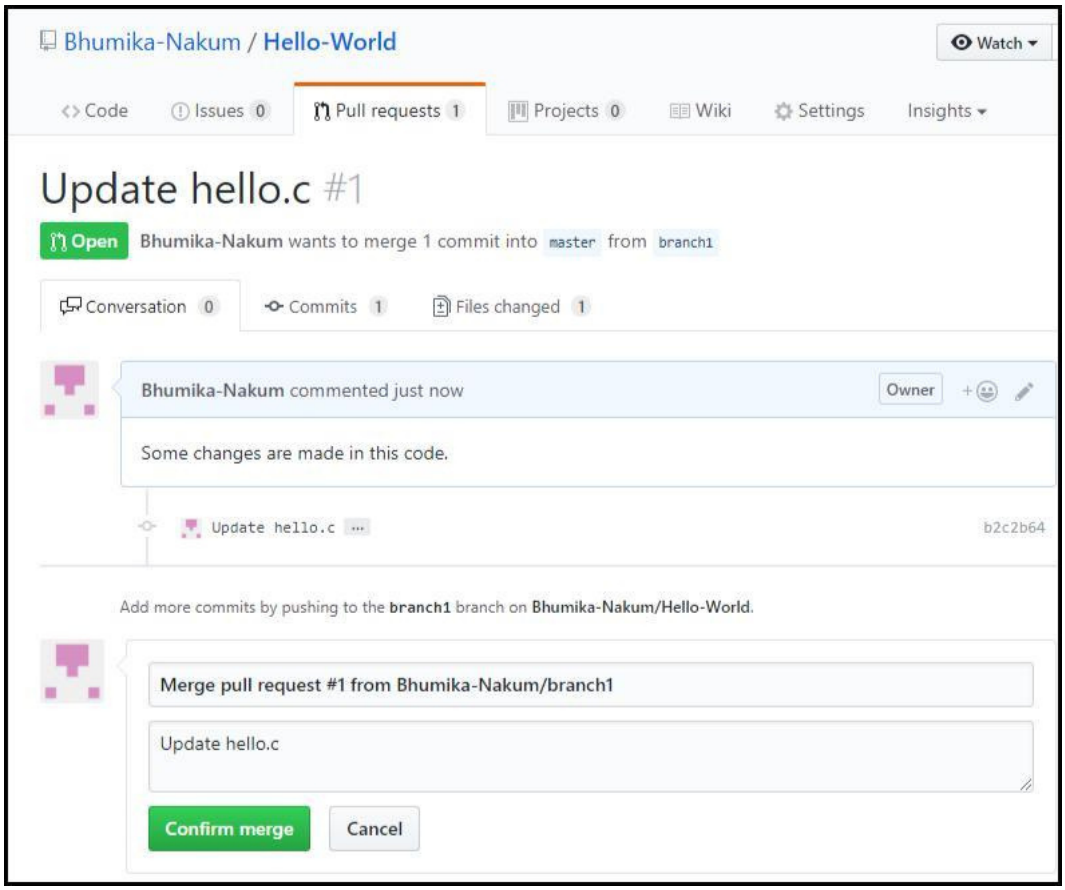
Step 18: The Pull Request will be opened and you can see the changes you have made in the file. The changes will be highlighted. Click on “Create pull request”.



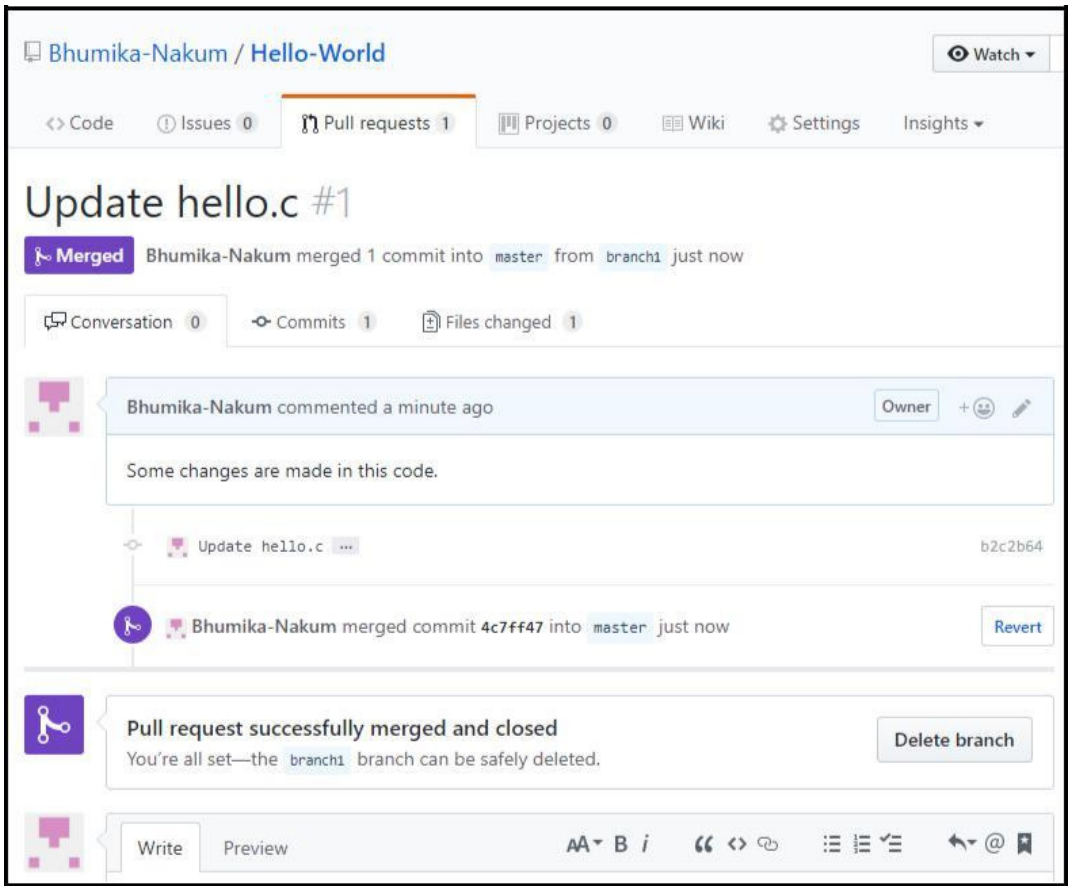
Step 19: Pull Request will be created successfully. Now you can Merge Pull Request. For this, click on “Merge pull request”.



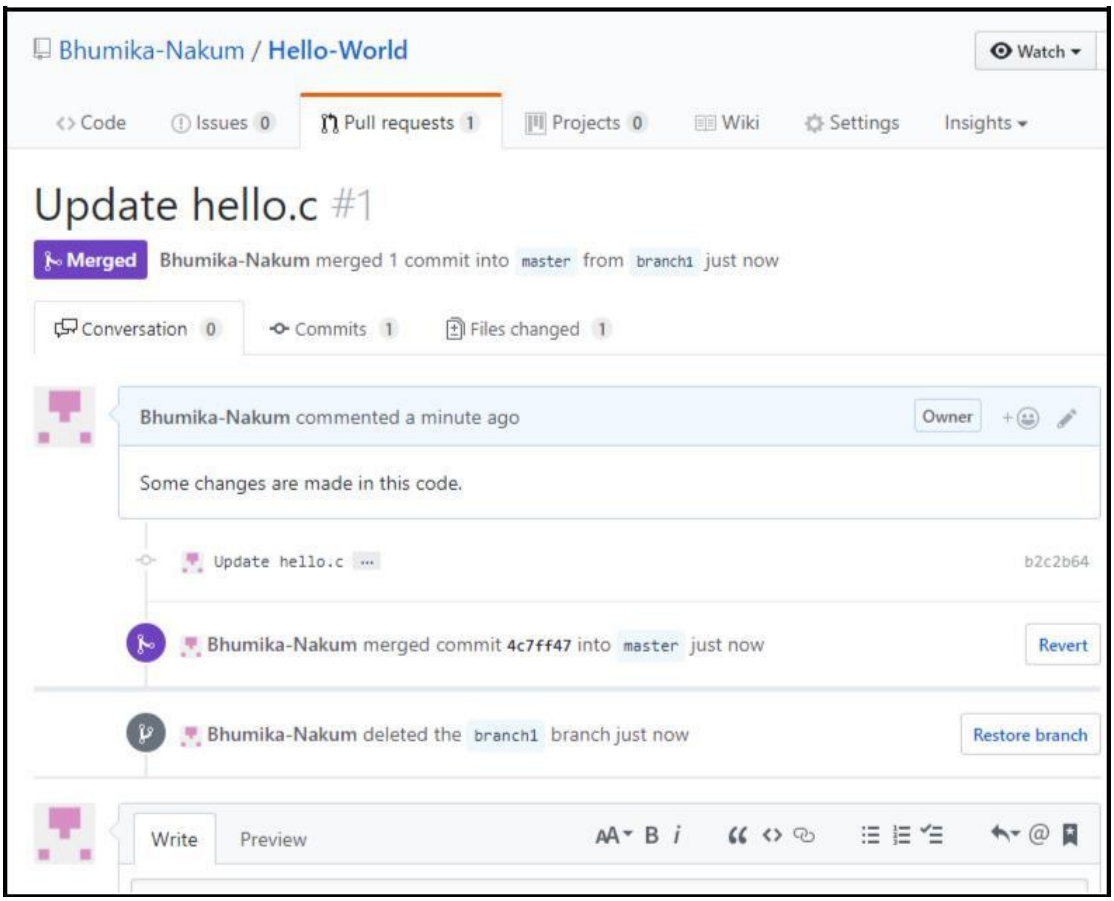
Step 20: Confirm the Merging by Clicking on “Confirm merge”.



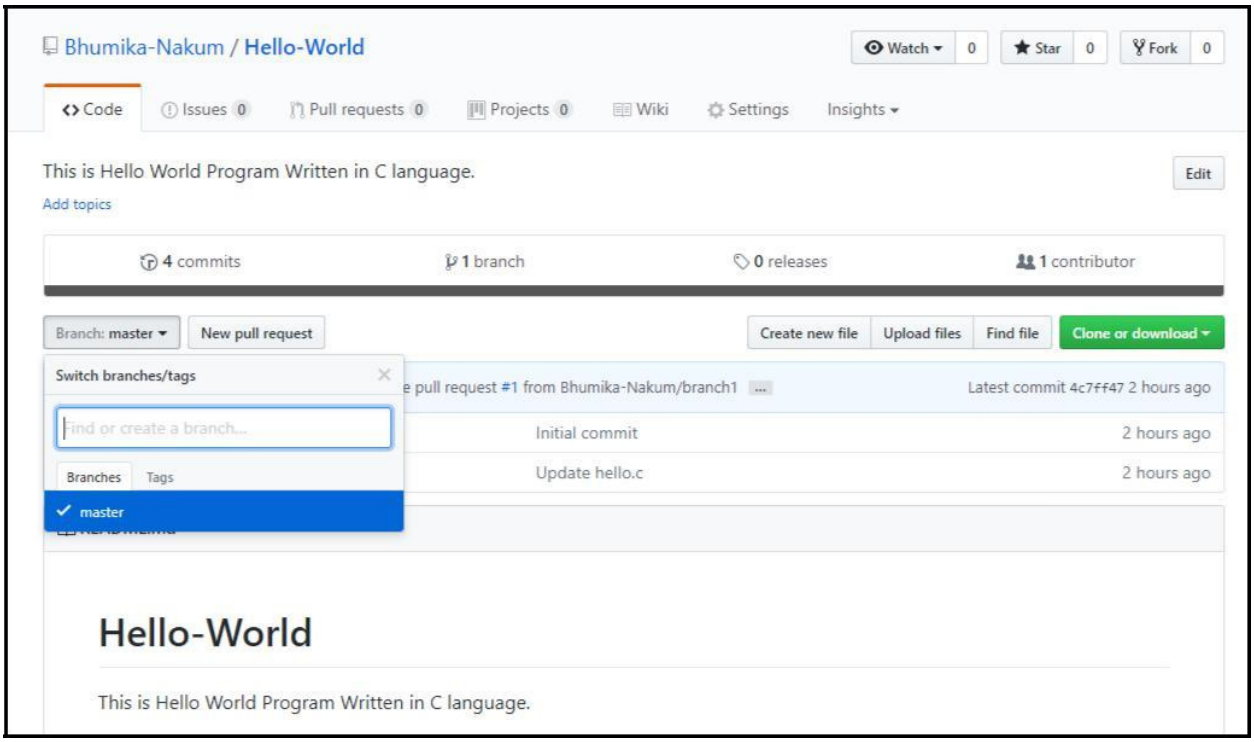
Step 21: After merging, let us delete branch1. Click on “Delete branch”.



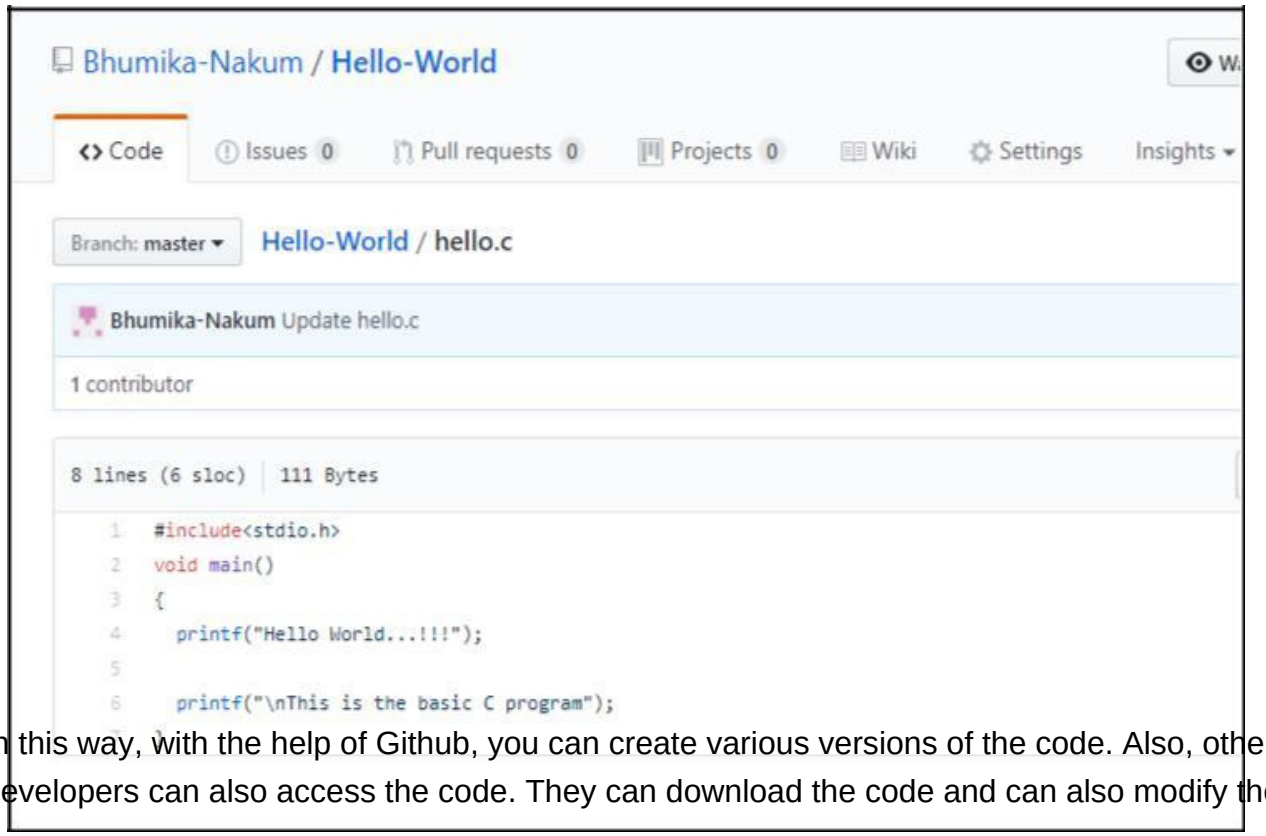
Step 22: After clicking “Delete branch”, branch1 will be deleted from your repository.



Step 23: Now, click on the name of the repository (HelloWorld). You can see that the branch1 is deleted from the repository.



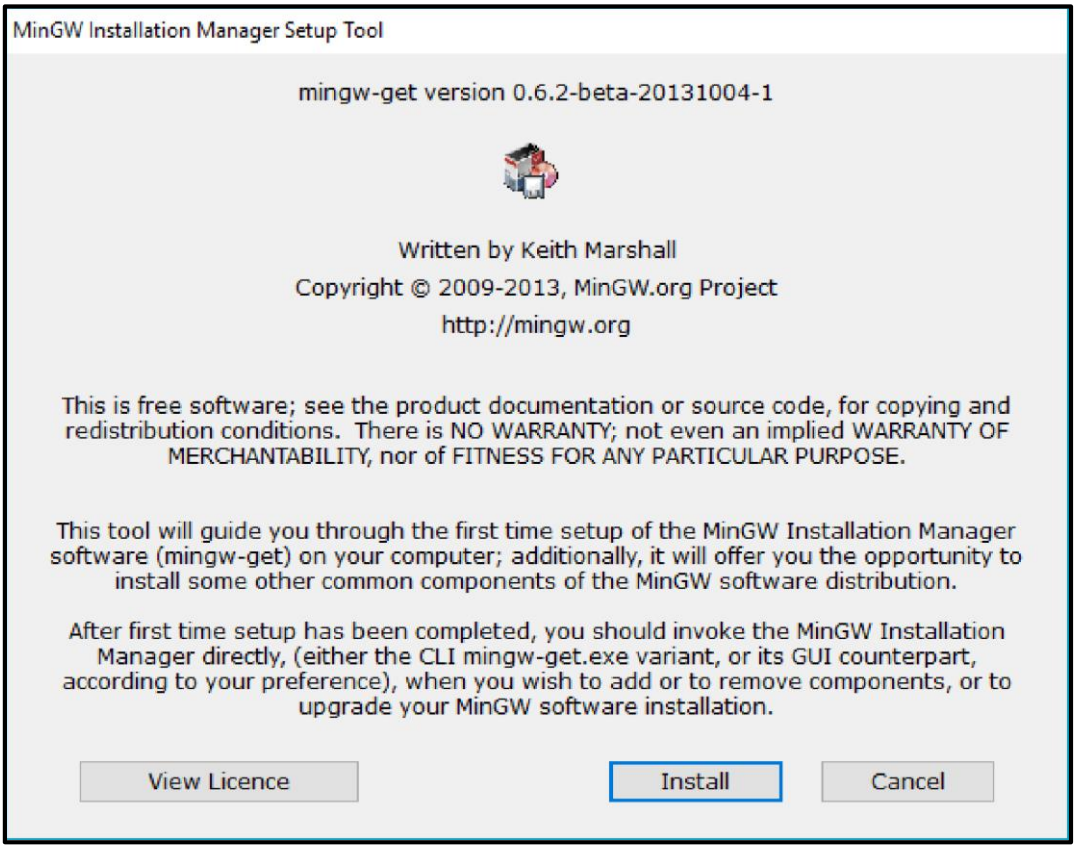
Step 24: Click on “hello.c” file. You can see that the “hello.c” file in the Master branch is modified and updated.



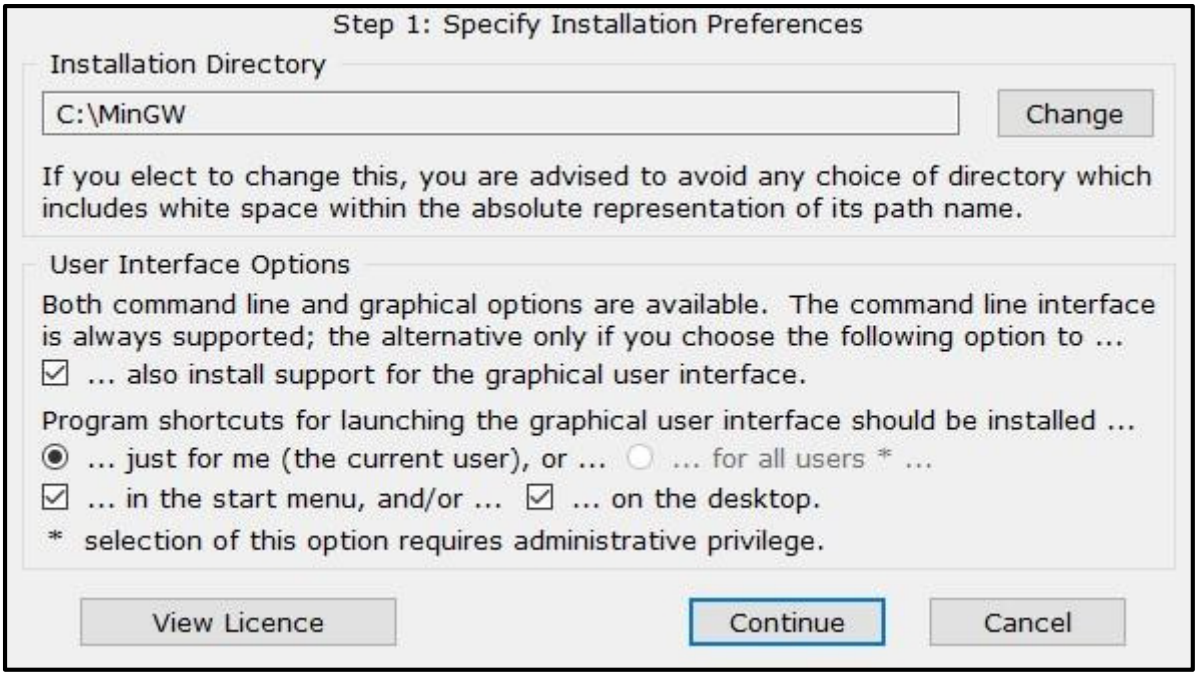
In this way, with the help of Github, you can create various versions of the code. Also, other developers can also access the code. They can download the code and can also modify the code.

Practical No. 5: Hands on with Open Source Software: GCC

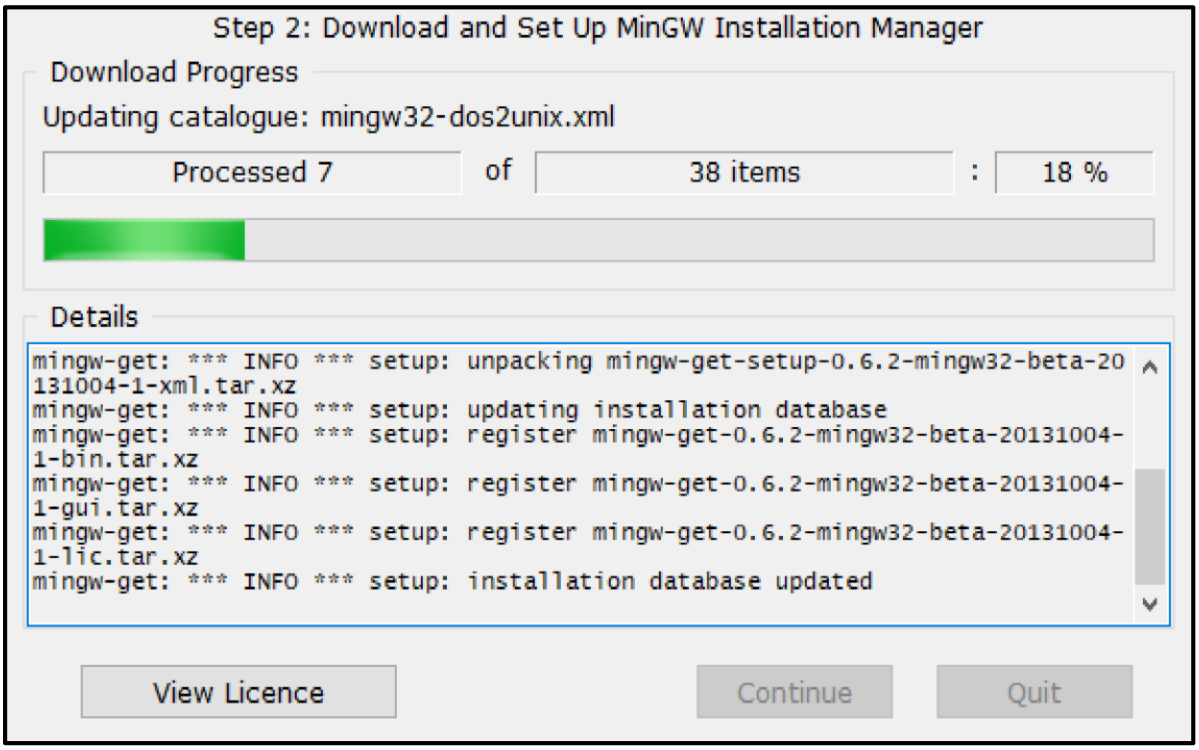
Step1: Download the software from https://sourceforge.net/projects/mingw/ link. Open the software and Click on “I nstall ”.



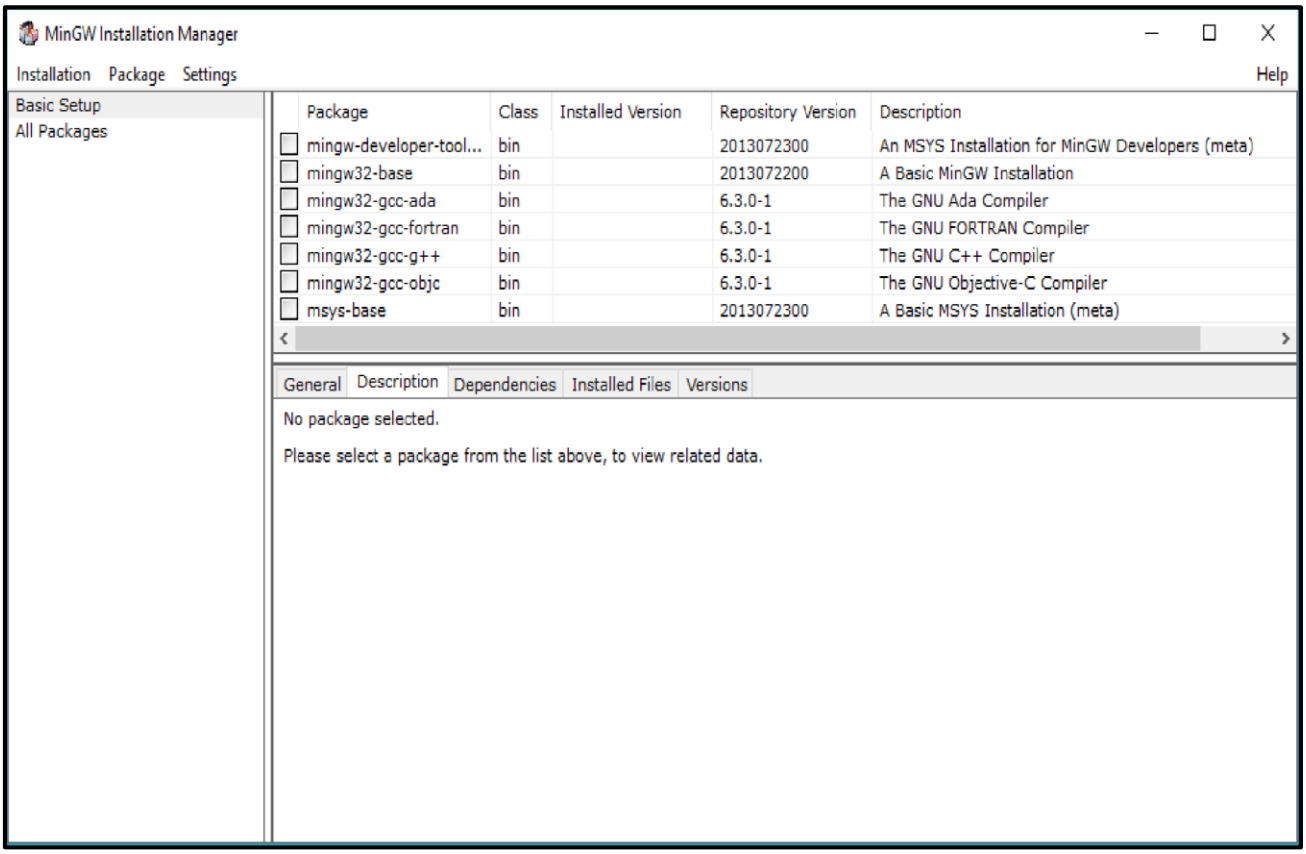
Step 2: Click on “C ontinue ”.



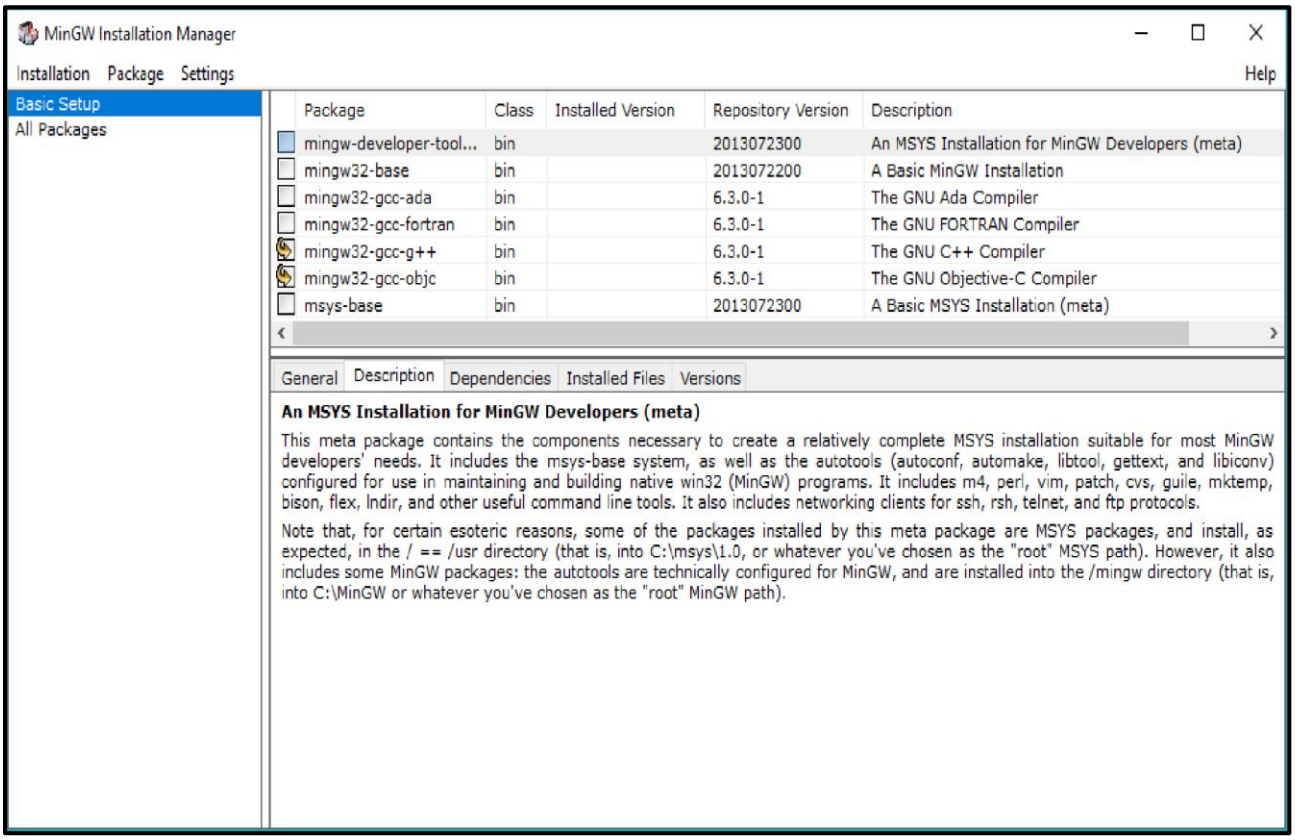
Step 3: After completion of installation click on “C ontinue ” .



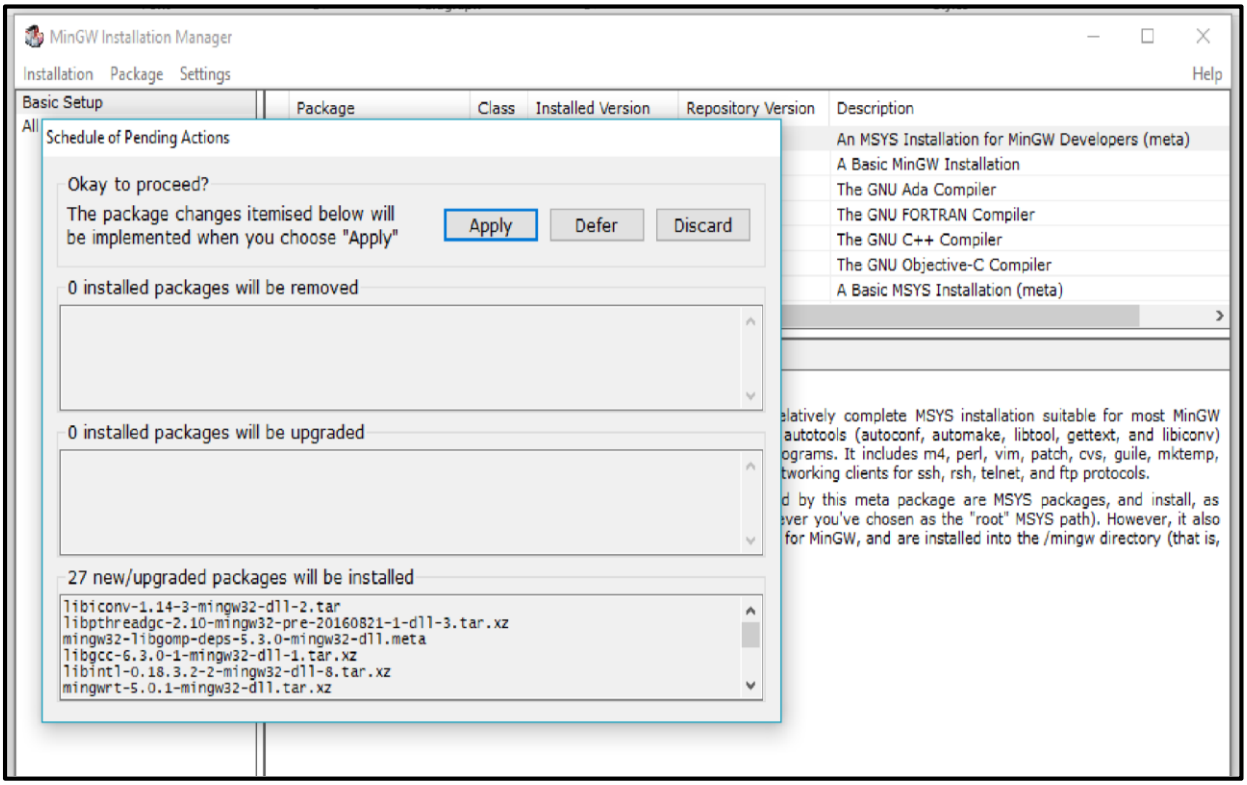
Step 4: MinGW Installation Manager window will be opened.



Step 5: Now mark C and C++ library files for Installation.

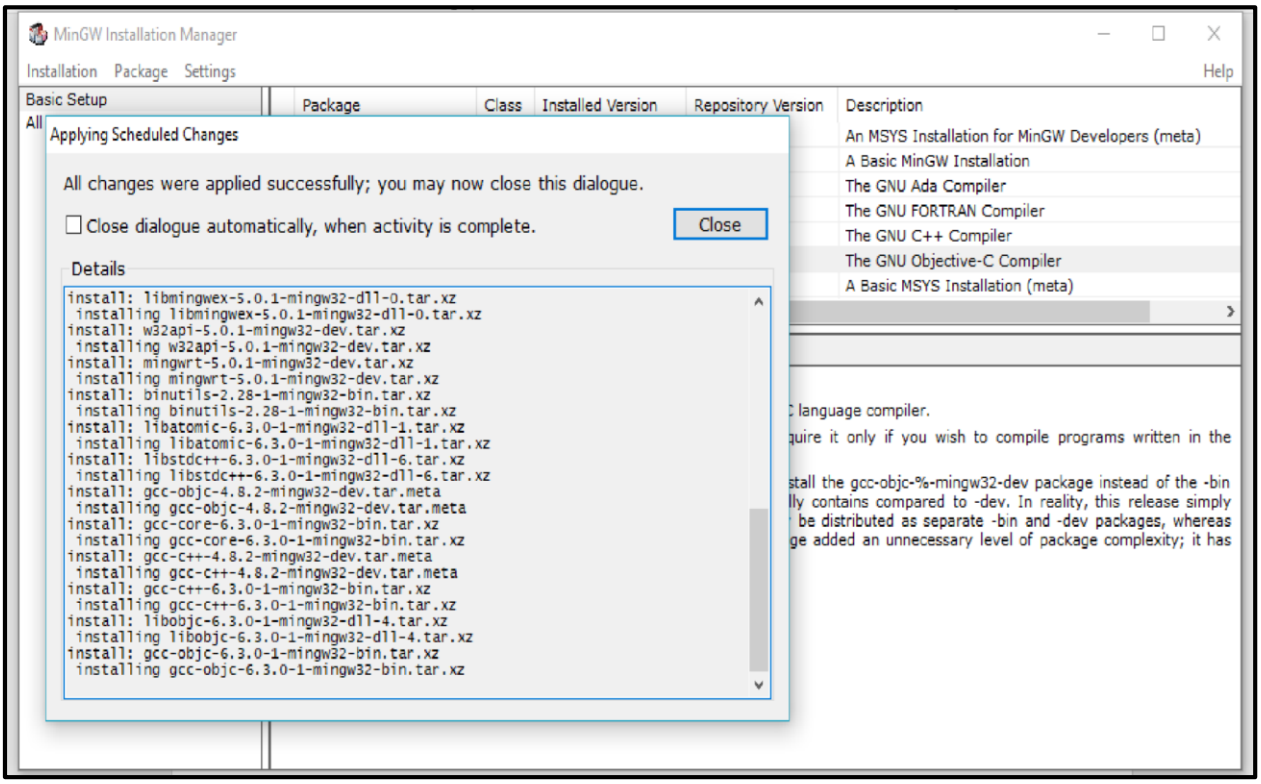


Step 6: Click on “I nstallation ” t ab and select “A pp ly Changes ”.Click on “A pply”.

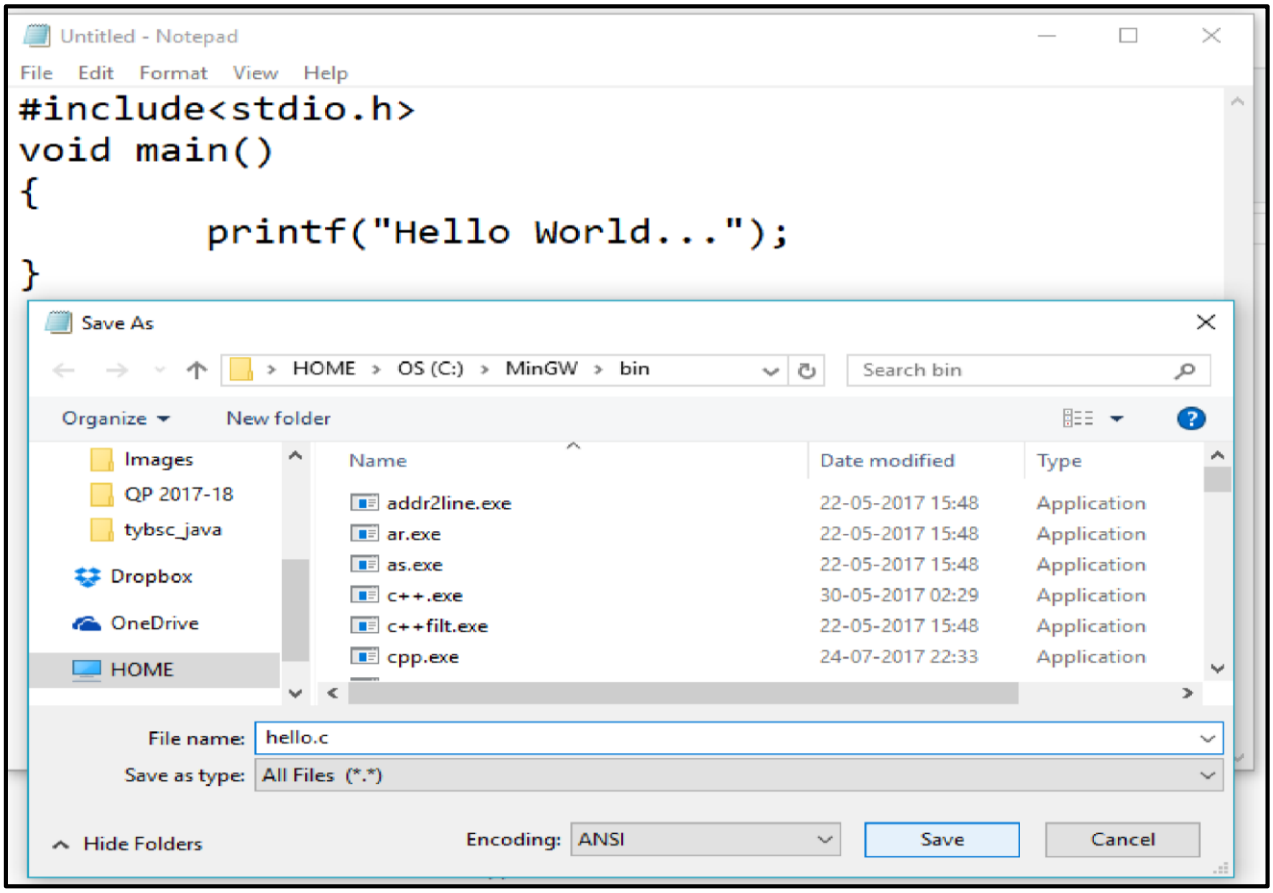


Step 7: The library files of C and C++ will be downloaded and installed. Now Click on

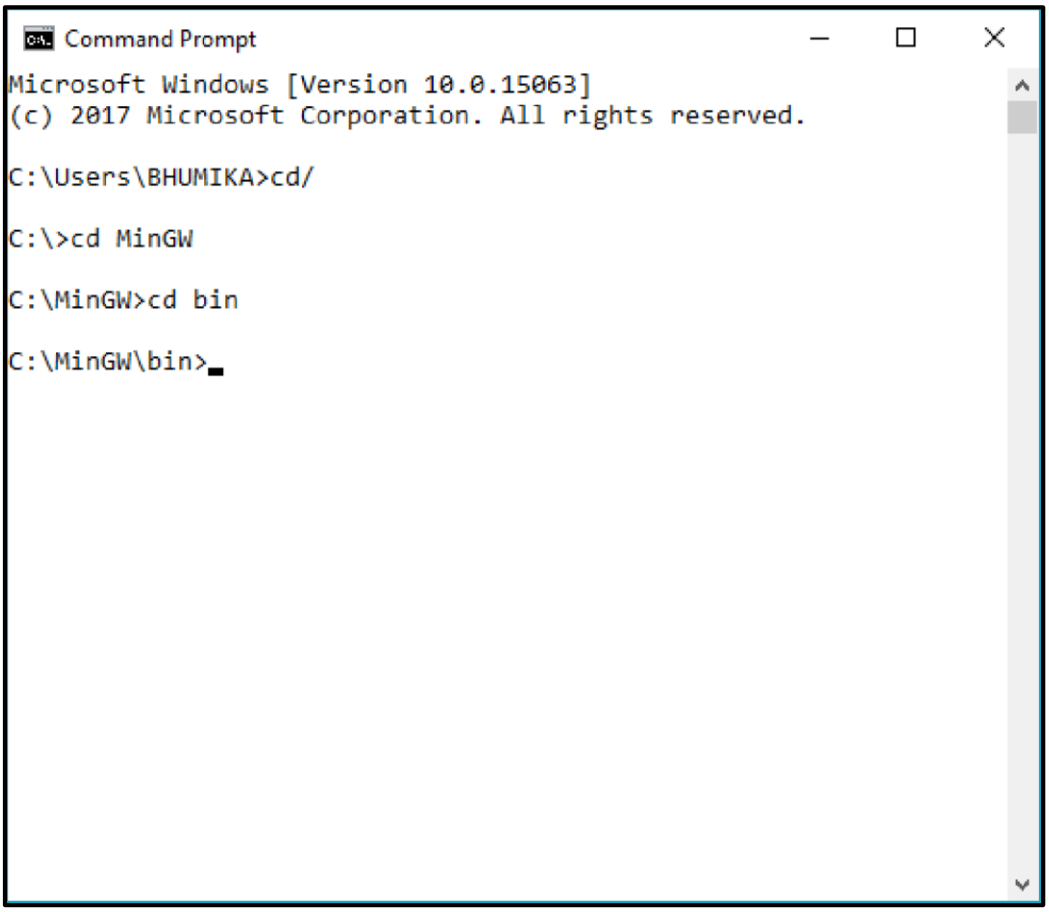
“C lose ”.



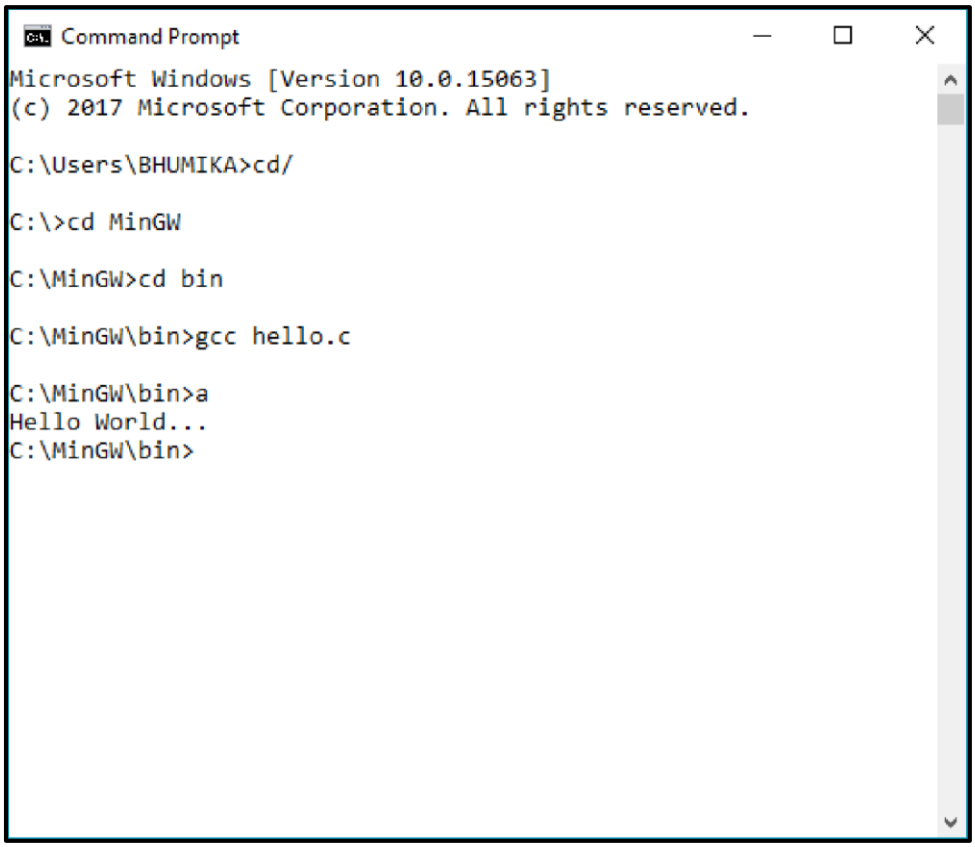
Step 8: Open Notepad and write C program and save the file as “hello.c” Save the file in C: \ MinGW\ bin



Step 9: Now open Command Prompt and type the following commands.



Step 10: After Setting the path, compile “hello.c” file by using “gcc” command and run the program by using “a” c ommand.



Practical 6: Hands On Open Source Software

Aim: Identify 5 open source software of your choice and write about them.

Practical View-Point:

1. Apache Server.

- The Apache HTTP Server, also called Apache, is a free and open-source cross-platform web server.
- Apache is developed and maintained by an open community of developers under the Apache Software Foundation.
- The vast majority of Apache HTTP Server instances run on a Linux distribution, but current versions also run on Windows and OS/2, and a wide variety of Unix-like systems.
- Past versions also ran on OpenVMS, NetWare and other operating systems.
- Originally based on the NCSA HTTPd server, development of Apache began in early 1995 after work on the NCSA code stalled.
- Apache played a key role in the initial growth of the World Wide Web, quickly overtaking NCSA HTTPd as the dominant HTTP server, and has remained most popular since April 1996.
- In 2009, it became the first web server software to serve more than 100 million websites.
- As of August 2018, it was estimated to serve 39% of all active websites and 35% of the top million websites.
- Apache server is secured, reliable and fast.
- It is the most widely used server software.

2. WordPress.

- WordPress is a free and open-source content management system (CMS) based on PHP and MySQL.
- Features include a plugin architecture and a template system.
- It is most associated with blogging, but supports other types of web content including more traditional mailing lists and forums, media galleries, and online stores.
- Used by more than 60 million websites, including 30.6% of the top 10 million websites as of April 2018, WordPress is the most popular website management system in use.
- WordPress has also been used for other application domains such as pervasive display systems (PDS).
- WordPress was released on May 27, 2003, by its founders, Matt Mullenweg and Mike Little. The software is released under the GPLv2 (or later) license.
- To function, WordPress has to be installed on a web server, either part of an Internet hosting service like WordPress.com or a computer running the software package WordPress.org in order to serve as a network host in its own right.
- A local computer may be used for single-user testing and learning purposes.

3. GCC.

 It stands for GNU Compiler Collection (GCC).

 It is a compiler system produced by the GNU Project supporting various programming languages.

 GCC is a key component of the GNU toolchain and the standard compiler for most Unix-like operating systems.

 The Free Software Foundation (FSF) distributes GCC under the GNU General Public License (GNU GPL).

 GCC has played an important role in the growth of free software, as both a tool and an example.

 Originally named the GNU C Compiler, when it only handled the C programming language, GCC 1.0 was released in 1987. It was extended to compile C++ in December of that year.

 Front ends were later developed for Objective-C, Objective-C++, Fortran, Java, Ada, and Go among others.

 GCC has been ported to a wide variety of instruction set architectures, and is widely deployed as a tool in the development of both free and proprietary software.

 GCC is also available for most embedded systems, including ARM-based; AMCC, and Freescale Power Architecture-based chips. The compiler can target a wide variety of platforms.

 As well as being the official compiler of the GNU operating system, GCC has been adopted as the standard compiler by many other modern Unix-like computer operating systems, including Linux and the BSD family, although FreeBSD and macOS have moved to the LLVM system.

 Versions are also available for Microsoft Windows and other operating systems; GCC can compile code for Android and iOS.

4. Mozilla Browser.

 Mozilla Firefox (or simply Firefox) is a free and open-source web browser developed by Mozilla Foundation and its subsidiary, Mozilla Corporation.

 Firefox is available for Windows, macOS, Linux, and BSD operating systems.

 Firefox uses the Gecko layout engine to render web pages, which implements current and anticipated web standards.

 Firefox for Android, is available for Android. An additional version, Firefox for iOS, was released on November 12, 2015, due to platform restrictions, it uses the WebKit layout engine instead of Gecko, as with all other iOS web browsers.

 Firefox was created in 2002 under the codename "Phoenix" by the Mozilla community members who desired a standalone browser, rather than the Mozilla Application Suite bundle.

 Firefox was released on November 9, 2004, and challenged Internet Explorer's dominance with 60 million downloads within nine months.

 Firefox usage grew to a peak of 32% at the end of 2009, temporarily making version 3.5 the world's most popular browser. Usage then declined in competition with Google Chrome.

 As of March 2018, Firefox has 11.6% usage share as a "desktop" browser, according to StatCounter, making it the second most popular such web browser; usage across all platforms is lower at 5.44%
 Firefox is still the most popular desktop browser in Cuba and Eritrea.
 According to Mozilla, as of December 2014, there were half a billion Firefox users around the world.

5. MySQL.

 MySQL is an open-source relational database management system (RDBMS).
 Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.
 The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements.
 MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.
 For proprietary use, several paid editions are available, and offer additional functionality.
 MySQL is a central component of the LAMP open-source web application software stack.
 LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python".
 Applications that use the MySQL database include: TYPO3, MODx, Joomla, WordPress, Simple Machines Forum, phpBB, MyBB, and Drupal.
 MySQL is also used in many high-profile, large-scale websites, including Google (though not for searches), Facebook, Twitter, Flickr, and YouTube.

6. GDB.

 The GNU Debugger (GDB) is a portable debugger that runs on many Unix-like systems and works for many programming languages, including Ada, C, C++, Objective-C, Free Pascal, Fortran, Go, Java and partially others.
 GDB was first written by Richard Stallman in 1986 as part of his GNU system, after his GNU Emacs was "reasonably stable".
 GDB is free software released under the GNU General Public License (GPL).
 It was modeled after the DBX debugger, which came with Berkeley Unix distributions.
 From 1990 to 1993 it was maintained by John Gilmore. Now it is maintained by the GDB Steering Committee which is appointed by the Free Software Foundation.
 GDB offers extensive facilities for tracing and altering the execution of computer programs.
 The user can monitor and modify the values of programs' internal variables, and even call functions independently of the program's normal behavior.
 GDB is still actively developed. As of version 7.0 new features include support for Python scripting and as of version 7.8 GNU Guile scripting as well.
 Since version 7.0, support for "reversible debugging" — allowing a debugging session to step backward, much like rewinding a crashed program to see what happened — is available.

- GDB offers a "remote" mode often used when debugging embedded systems. Remote operation is when GDB runs on one machine and the program being debugged runs on another.
- The debugger does not contain its own graphical user interface, and defaults to a command-line interface. Several front-ends have been built for it, such as UltraGDB, Xxgdb, Data Display Debugger (DDD) etc.

Practical 7

Aim: Open source Virtualization.

a) Evolution of Virtualization:

Virtualization is the creation of a virtual -- rather than actual -- version of something, such as an operating system, a server, a storage device or network resources.

Operating system virtualization is the use of software to allow a piece of hardware to run multiple operating system images at the same time. The technology got its start on mainframes decades ago, allowing administrators to avoid wasting expensive processing power.

Over the past decade there has been a great deal of work on improving the performance, enhancing the flexibility, and increasing the manageability of virtualization technologies.

Developments in the past five years alone, for example, include the ability to move a running virtual machine, along with its live operating system and applications, to a physical host without major downtime. The industry has also recently witnessed the ability of virtualization to log the actions of a virtual machine in real time, with the purpose of being able to roll back an entire system to an arbitrary point and then roll it forward for debugging or auditing. These and other recent developments have positioned virtualization as a core technology in cloud computing and have facilitated the technology’s move to the desktop.

b) How Virtualization works:

Virtualization describes a technology in which an application, guest operating system or data storage is abstracted away from the true underlying hardware or software. A key use of virtualization technology is server virtualization, which uses a software layer called a hypervisor to emulate the underlying hardware. This often includes the CPU's memory, I/O and network traffic. The guest operating system, normally interacting with true hardware, is now doing so with a software emulation of that hardware, and often the guest operating system has no idea it's on virtualized hardware. While the performance of this virtual system is not equal to the performance of the operating system running on true hardware, the concept of virtualization works because most guest operating systems and applications don't need the full use of the underlying hardware. This allows for greater flexibility, control and isolation by removing the dependency on a given hardware platform. While initially meant for server virtualization, the concept of virtualization has spread to applications, networks, data and desktops.

c) Types of Virtualization:

There are six areas of IT where virtualization is making headway:

a) Network virtualization is a method of combining the available resources in a network by splitting up the available bandwidth into channels, each of which is independent from the others and can be assigned -- or reassigned -- to a particular server or device in real time. The idea is that virtualization disguises the true complexity of the network by separating it into manageable parts; much like your partitioned hard drive makes it easier to manage your files.

b) Storage virtualization is the pooling of physical storage from multiple network storage devices into what appears to be a single storage device that is managed from a central console. Storage virtualization is commonly used in storage area networks.

c) Server virtualization is the masking of server resources -- including the number and identity of individual physical servers, processors and operating systems -- from server users. The intention is to spare the user from having to understand and manage complicated details of server resources while increasing resource sharing and utilization and maintaining the capacity to expand later.

The layer of software that enables this abstraction is often referred to as the hypervisor. The most common hypervisor -- Type 1 -- is designed to sit directly on bare metal and provide the ability to virtualize the hardware platform for use by the virtual machines (VMs). KVM virtualization is a Linux kernel-based virtualization hypervisor that provides Type 1 virtualization benefits similar to other hypervisors. KVM is licensed under open source. A Type 2 hypervisor requires a host operating system and is more often used for testing/labs.

d) Data virtualization is abstracting the traditional technical details of data and data management, such as location, performance or format, in favor of broader access and more resiliencies tied to business needs.

e) Desktop virtualization is virtualizing a workstation load rather than a server. This allows the user to access the desktop remotely, typically using a thin client at the desk. Since the workstation is essentially running in a data center server, access to it can be both more secure and portable. The operating system license does still need to be accounted for as well as the infrastructure.

f) Application virtualization is abstracting the application layer away from the operating system. This way the application can run in an encapsulated form without being depended upon on the operating system underneath. This can allow a Windows application to run on Linux and vice versa, in addition to adding a level of isolation.

PRACTICAL 8

AIM :Open Source Containerization

1) What is containerization

Containerization is a type of virtualization strategy that emerged as an alternative to traditional hypervisor-based virtualization. As with the latter, container-based virtualization involves creating specific virtual pieces of a hardware infrastructure, but unlike the traditional approach, which fully splits these virtual machines from the rest of the architecture, containerization just creates separate containers at the operating system level.

In containerization, the operating system is shared by the different containers rather than cloned for each virtual machine. The open source Docker provides a container virtualization platform that serves as a good alternative to hypervisor-based arrangements.

Containerization has also emerged as a potential solution to mobile security problems for multi-use phones or mobile devices such as in the trend called "bring your own device" (BYOD), wherein companies allow employees to use their personal devices for work. However, security experts point out that, although containerization could work to wall off sensitive corporate data on a multi-use phone, it would not work against jail breaking or certain kinds of vulnerabilities inherent in the mobile device operating system itself.

2) Difference between containerization and virtualization

VIRTUALIZATION

A virtual machine mimics a complete server. In a typical virtualized server, each VM "guest" includes a complete operating system along with any drivers, binaries or libraries, and then the actual application.
Each VM then runs atop a hypervisor, which itself runs on a host operating system and in turn operates the physical server hardware.
It's a tried-and-true approach, but it's also easy to see how each iteration of the guest operating system and supporting binaries can cause duplication between VMs; it wastes precious server memory, which limits the number of VMs that each server can support.

CONTAINERIZATION

The concept of containerization basically allows virtual instances to share a single host operating system and relevant binaries, libraries or drivers.
This approach reduces wasted resources because each container only holds the application and related binaries or libraries.
Containers use the same host operating system (OS) repeatedly, instead of installing (and paying to license) an OS for each guest VM.
This is often referred to as operating system-level virtualization. The role of a hypervisor is instead handled by a containerization engine, like Docker, which installs atop the host operating system.

3) Example of containerization

Container Linux (formerly Core OS Linux) — one of the first lightweight container operating systems built for containers

(a) RancherOS— a simplified Linux distribution built from containers, specifically for running containers.

(b) Photon OS— a minimal Linux container host, optimized to run on VMware platforms.

(c) Project Atomic Host— Red Hat's lightweight container OS has versions that are based on CentOS and Fedora, and there is also a downstream enterprise version in Red Hat Enterprise Linux.

(d) Ubuntu Core— the smallest Ubuntu version, Ubuntu Core is designed as a host operating system for IoT devices and large-scale cloud container deployments