1. What is free software?

**“Free software” means software that respects users' freedom and community. Roughly, it means that the users have the freedom to run, copy, distribute, study, change and improve the software. Thus, “free software” is a matter of liberty, not price. To understand the concept, you should think of “free” as in “free speech,” not as in “free beer”. We sometimes call it “libre software” to show we do not mean it is gratis.**

**We campaign for these freedoms because everyone deserves them. With these freedoms, the users (both individually and collectively) control the program and what it does for them. When users don't control the program, we call it a “nonfree” or “proprietary” program. The nonfree program controls the users, and the developer controls the program; this makes the program an instrument of unjust power.**

**A program is free software if the program's users have the four essential freedoms:**

**The freedom to run the program as you wish, for any purpose (freedom 0).**

**The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.**

**The freedom to redistribute copies so you can help your neighbor (freedom 2).**

**The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.**

**A program is free software if it gives users adequately all of these freedoms. Otherwise, it is nonfree. While we can distinguish various nonfree distribution schemes in terms of how far they fall short of being free, we consider them all equally unethical.**

**In any given scenario, these freedoms must apply to whatever code we plan to make use of, or lead others to make use of. For instance, consider a program A which automatically launches a program B to handle some cases. If we plan to distribute A as it stands, that implies users will need B, so we need to judge whether both A and B are free. However, if we plan to modify A so that it doesn't use B, only A needs to be free; we can ignore B.**

**The rest of this page clarifies certain points about what makes specific freedoms adequate or not.**

**Freedom to distribute (freedoms 2 and 3) means you are free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere. Being free to do these things means (among other things) that you do not have to ask or pay for permission to do so.**

**You should also have the freedom to make modifications and use them privately in your own work or play, without even mentioning that they exist. If you do publish your changes, you should not be required to notify anyone in particular, or in any particular way.**

**The freedom to run the program means the freedom for any kind of person or organization to use it on any kind of computer system, for any kind of overall job and purpose, without being required to communicate about it with the developer or any other specific entity. In this freedom, it is the user's purpose that matters, not the developer's purpose; you as a user are free to run the program for your purposes, and if you distribute it to someone else, she is then free to run it for her purposes, but you are not entitled to impose your purposes on her.**

**The freedom to run the program as you wish means that you are not forbidden or stopped from doing so. It has nothing to do with what functionality the program has, or whether it is useful for what you want to do.**

**The freedom to redistribute copies must include binary or executable forms of the program, as well as source code, for both modified and unmodified versions. (Distributing programs in runnable form is necessary for conveniently installable free operating systems.) It is OK if there is no way to produce a binary or executable form for a certain program (since some languages don't support that feature), but you must have the freedom to redistribute such forms should you find or develop a way to make them.**

**In order for freedoms 1 and 3 (the freedom to make changes and the freedom to publish the changed versions) to be meaningful, you must have access to the source code of the program. Therefore, accessibility of source code is a necessary condition for free software. Obfuscated “source code” is not real source code and does not count as source code.**

**Freedom 1 includes the freedom to use your changed version in place of the original. If the program is delivered in a product designed to run someone else's modified versions but refuse to run yours — a practice known as “tivoization” or “lockdown”, or (in its practitioners' perverse terminology) as “secure boot” — freedom 1 becomes an empty pretense rather than a practical reality. These binaries are not free software even if the source code they are compiled from is free.**

**One important way to modify a program is by merging in available free subroutines and modules. If the program's license says that you cannot merge in a suitably licensed existing module — for instance, if it requires you to be the copyright holder of any code you add — then the license is too restrictive to qualify as free.**

**Freedom 3 includes the freedom to release your modified versions as free software. A free license may also permit other ways of releasing them; in other words, it does not have to be a copyleft license. However, a license that requires modified versions to be nonfree does not qualify as a free license.**

**In order for these freedoms to be real, they must be permanent and irrevocable as long as you do nothing wrong; if the developer of the software has the power to revoke the license, or retroactively add restrictions to its terms, without your doing anything wrong to give cause, the software is not free.**

**However, certain kinds of rules about the manner of distributing free software are acceptable, when they don't conflict with the central freedoms. For example, copyleft (very simply stated) is the rule that when redistributing the program, you cannot add restrictions to deny other people the central freedoms. This rule does not conflict with the central freedoms; rather it protects them.**

**In the GNU project, we use copyleft to protect the four freedoms legally for everyone. We believe there are important reasons why it is better to use copyleft. However, noncopylefted free software is ethical too. See Categories of Free Software for a description of how “free software,” “copylefted software” and other categories of software relate to each other.**

**“Free software” does not mean “noncommercial”. A free program must be available for commercial use, commercial development, and commercial distribution. Commercial development of free software is no longer unusual; such free commercial software is very important. You may have paid money to get copies of free software, or you may have obtained copies at no charge. But regardless of how you got your copies, you always have the freedom to copy and change the software, even to sell copies.**

**Whether a change constitutes an improvement is a subjective matter. If your right to modify a program is limited, in substance, to changes that someone else considers an improvement, that program is not free.**

**However, rules about how to package a modified version are acceptable, if they don't substantively limit your freedom to release modified versions, or your freedom to make and use modified versions privately. Thus, it is acceptable for the license to require that you change the name of the modified version, remove a logo, or identify your modifications as yours. As long as these requirements are not so burdensome that they effectively hamper you from releasing your changes, they are acceptable; you're already making other changes to the program, so you won't have trouble making a few more.**

**Rules that “if you make your version available in this way, you must make it available in that way also” can be acceptable too, on the same condition. An example of such an acceptable rule is one saying that if you have distributed a modified version and a previous developer asks for a copy of it, you must send one. (Note that such a rule still leaves you the choice of whether to distribute your version at all.) Rules that require release of source code to the users for versions that you put into public use are also acceptable.**

**A special issue arises when a license requires changing the name by which the program will be invoked from other programs. That effectively hampers you from releasing your changed version so that it can replace the original when invoked by those other programs. This sort of requirement is acceptable only if there's a suitable aliasing facility that allows you to specify the original program's name as an alias for the modified version.**

**Sometimes government export control regulations and trade sanctions can constrain your freedom to distribute copies of programs internationally. Software developers do not have the power to eliminate or override these restrictions, but what they can and must do is refuse to impose them as conditions of use of the program. In this way, the restrictions will not affect activities and people outside the jurisdictions of these governments. Thus, free software licenses must not require obedience to any nontrivial export regulations as a condition of exercising any of the essential freedoms.**

**Merely mentioning the existence of export regulations, without making them a condition of the license itself, is acceptable since it does not restrict users. If an export regulation is actually trivial for free software, then requiring it as a condition is not an actual problem; however, it is a potential problem, since a later change in export law could make the requirement nontrivial and thus render the software nonfree.**

**A free license may not require compliance with the license of a nonfree program. Thus, for instance, if a license requires you to comply with the licenses of “all the programs you use”, in the case of a user that runs nonfree programs this would require compliance with the licenses of those nonfree programs; that makes the license nonfree.**

**It is acceptable for a free license to specify which jurisdiction's law applies, or where litigation must be done, or both.**

**Most free software licenses are based on copyright, and there are limits on what kinds of requirements can be imposed through copyright. If a copyright-based license respects freedom in the ways described above, it is unlikely to have some other sort of problem that we never anticipated (though this does happen occasionally). However, some free software licenses are based on contracts, and contracts can impose a much larger range of possible restrictions. That means there are many possible ways such a license could be unacceptably restrictive and nonfree.**

**We can't possibly list all the ways that might happen. If a contract-based license restricts the user in an unusual way that copyright-based licenses cannot, and which isn't mentioned here as legitimate, we will have to think about it, and we will probably conclude it is nonfree.**

**When talking about free software, it is best to avoid using terms like “give away” or “for free,” because those terms imply that the issue is about price, not freedom. Some common terms such as “piracy” embody opinions we hope you won't endorse. See Confusing Words and Phrases that are Worth Avoiding for a discussion of these terms. We also have a list of proper translations of “free software” into various languages.**

**Finally, note that criteria such as those stated in this free software definition require careful thought for their interpretation. To decide whether a specific software license qualifies as a free software license, we judge it based on these criteria to determine whether it fits their spirit as well as the precise words. If a license includes unconscionable restrictions, we reject it, even if we did not anticipate the issue in these criteria. Sometimes a license requirement raises an issue that calls for extensive thought, including discussions with a lawyer, before we can decide if the requirement is acceptable. When we reach a conclusion about a new issue, we often update these criteria to make it easier to see why certain licenses do or don't qualify.**

1. When was free software born?

**Founded by Richard Stallman, 1983**

1. When was proprietary software born?

**In 1969, IBM, under threat of antitrust litigation, led an industry change by starting to charge separately for (mainframe) software and services. Bill Gates’ “Open Letter to Hobbyists” was written in 1976 and Gates was a leader in proprietary software**

1. What is the Free Software Movement and who started it?

**A social movement with the goal of obtaining and guaranteeing certain freedoms for software users, namely the freedom to run the software, to study and change the software, and to redistribute copies with or without changes. Although drawing on traditions and philosophies among members of the 1970s hacker culture and academia, Richard Stallman formally founded the movement in 1983 by launching the GNU Project. The Free Software Foundation was founded in 1885**

1. What is the difference between free and open source software (see Open Source Definition)?

**1. Free Redistribution**

**The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.**

**Rationale: By constraining the license to require free redistribution, we eliminate the temptation for licensors to throw away many long-term gains to make short-term gains. If we didn't do this, there would be lots of pressure for cooperators to defect.**

**2. Source Code**

**The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.**

**Rationale: We require access to un-obfuscated source code because you can't evolve programs without modifying them. Since our purpose is to make evolution easy, we require that modification be made easy.**

**3. Derived Works**

**The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.**

**Rationale: The mere ability to read source isn't enough to support independent peer review and rapid evolutionary selection. For rapid evolution to happen, people need to be able to experiment with and redistribute modifications.**

**4. Integrity of The Author's Source Code**

**The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.**

**Rationale: Encouraging lots of improvement is a good thing, but users have a right to know who is responsible for the software they are using. Authors and maintainers have reciprocal right to know what they're being asked to support and protect their reputations.**

**Accordingly, an open-source license must guarantee that source be readily available, but may require that it be distributed as pristine base sources plus patches. In this way, "unofficial" changes can be made available but readily distinguished from the base source.**

**5. No Discrimination Against Persons or Groups**

**The license must not discriminate against any person or group of persons.**

**Rationale: In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open sources. Therefore we forbid any open-source license from locking anybody out of the process.**

**Some countries, including the United States, have export restrictions for certain types of software. An OSD-conformant license may warn licensees of applicable restrictions and remind them that they are obliged to obey the law; however, it may not incorporate such restrictions itself.**

**6. No Discrimination Against Fields of Endeavor**

**The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.**

**Rationale: The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.**

**7. Distribution of License**

**The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.**

**Rationale: This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.**

**8. License Must Not Be Specific to a Product**

**The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.**

**Rationale: This clause forecloses yet another class of license traps.**

**9. License Must Not Restrict Other Software**

**The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.**

**Rationale: Distributors of open-source software have the right to make their own choices about their own software.**

**Yes, the GPL v2 and v3 are conformant with this requirement. Software linked with GPLed libraries only inherits the GPL if it forms a single work, not any software with which they are merely distributed.**

**10. License Must Be Technology-Neutral**

**No provision of the license may be predicated on any individual technology or style of interface.**

**A program is free software, for you, a particular user, if:**

**You have the freedom to run the program as you wish, for any purpose.**

**You have the freedom to modify the program to suit your needs. (To make this freedom effective in practice, you must have access to the source code, since making changes in a program without having the source code is exceedingly difficult.)**

**You have the freedom to redistribute copies, either gratis or for a fee.**

**You have the freedom to distribute modified versions of the program, so that the community can benefit from your improvements.**

**Since “free” refers to freedom, not to price, there is no contradiction between selling copies and free software. In fact, the freedom to sell copies is crucial: collections of free software sold on CD-ROMs are important for the community, and selling them is an important way to raise funds for free software development. Therefore, a program which people are not free to include on these collections is not free software.**

**Because of the ambiguity of “free”, people have long looked for alternatives, but no one has found a better term. The English language has more words and nuances than any other, but it lacks a simple, unambiguous, word that means “free”, as in freedom—“unfettered” being the word that comes closest in meaning. Such alternatives as “liberated”, “freedom”, and “open” have either the wrong meaning or some other disadvantage.**

1. What was the first free operating system?
2. Does free software have a license?

**Yes.**

1. What is “copy left” and how is it different than copyright?

**Its copyright flipped over. Software is copyrighted but gives permission to redistribute software under particular terms, no more no less.**

1. What was the first license for free software?

**GNU General Public Liscense**

1. How can one make money from free open source software?

**Sell Support Contracts, Sell Value-Added Enhancements, Sell Documentation, Sell Binaries, Sell Your Expertise as a Consultant**

1. Which was the first free operating system kernel?

**Linux**

1. Why was the Linux kernel released before the GNU Hurd kernel?
2. Why did Linus Torvalds start to develop the Linux kernel?
3. What was the role of Apache Web Server in the adoption of Linux?

**To build complex web applications.**

1. Why does open source software work (“The Cathedral and the Bazaar”)?

**A peer-to-peer, market style, environment that supports better review.**

1. What are some factors that influence the success of open source software?

**With the rapid rise in the use of Open Source Software (OSS) in all types of applications, it is important to know which factors can lead to OSS success. OSS projects evolve and transform over time; therefore success must be examined longitudinally over a period of time. In this research, we examine two measures of project success: project popularity and developer activity, of 283 OSS projects over a span of 3 years, in order to observe changes over time. A comprehensive research model of OSS success is developed which includes both extrinsic and intrinsic attributes. Results show that while many of the hypothesized relationships are supported, there were marked differences in some of the relationships at different points in time lending support to the notion that different factors need to be emphasized as the OSS project unfolds over time.**

1. What is Linus’s Law?

**Linus's Law is a claim about software development, named in honor of Linus Torvalds and formulated by Eric S. Raymond in his essay and book The Cathedral and the Bazaar (1999).[1][2] The law states that "given enough eyeballs, all bugs are shallow"; or more formally: "Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix will be obvious to someone." Presenting the code to multiple developers with the purpose of reaching consensus about its acceptance is a simple form of software reviewing. Researchers and practitioners have repeatedly shown the effectiveness of various types of reviewing process in finding bugs and security**

1. When is Windows Refund day?

**February 15th 1999 was windows refund day. Open source OS users went to microsoft's offices to return their unused licenses of windows that they were forced to acquire since they were bundled with the machine they bought.**