```
// MEGA is available under the terms of a GNU licence. see below:
//
//                    GNU GENERAL PUBLIC LICENSE
//                       Version 3, 29 June 2007

// Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
// Everyone is permitted to copy and distribute verbatim copies
// of this license document, but changing it is not allowed.

//                            Preamble

//  The GNU General Public License is a free, copyleft license for
//software and other kinds of works.

//  The licenses for most software and other practical works are designed
//to take away your freedom to share and change the works.  By contrast,
//the GNU General Public License is intended to guarantee your freedom to
//share and change all versions of a program--to make sure it remains free
//software for all its users.  We, the Free Software Foundation, use the
//GNU General Public License for most of our software; it applies also to
//any other work released this way by its authors.  You can apply it to
//your programs, too.

//  When we speak of free software, we are referring to freedom, not
//price.  Our General Public Licenses are designed to make sure that you
//have the freedom to distribute copies of free software (and charge for
//them if you wish), that you receive source code or can get it if you
//want it, that you can change the software or use pieces of it in new
//free programs, and that you know you can do these things.

//  To protect your rights, we need to prevent others from denying you
//these rights or asking you to surrender the rights.  Therefore, you have
//certain responsibilities if you distribute copies of the software, or if
//you modify it: responsibilities to respect the freedom of others.

//  For example, if you distribute copies of such a program, whether
//gratis or for a fee, you must pass on to the recipients the same
//freedoms that you received.  You must make sure that they, too, receive
//or can get the source code.  And you must show them these terms so they
//know their rights.

//  Developers that use the GNU GPL protect your rights with two steps:
//() assert copyright on the software, and (2) offer you this License
//giving you legal permission to copy, distribute and/or modify it.

//  For the developers' and authors' protection, the GPL clearly explains
//that there is no warranty for this free software.  For both users' and
//authors' sake, the GPL requires that modified versions be marked as
//changed, so that their problems will not be attributed erroneously to
//authors of previous versions.

//  Some devices are designed to deny users access to install or run
//modified versions of the software inside them, although the manufacturer
//can do so.  This is fundamentally incompatible with the aim of
//protecting users' freedom to change the software.  The systematic
//pattern of such abuse occurs in the area of products for individuals to
//use, which is precisely where it is most unacceptable.  Therefore, we
//have designed this version of the GPL to prohibit the practice for those
//products.  If such problems arise substantially in other domains, we
//stand ready to extend this provision to those domains in future versions
//of the GPL, as needed to protect the freedom of users.

//  Finally, every program is threatened constantly by software patents.
//States should not allow patents to restrict development and use of
//software on general-purpose computers, but in those that do, we wish to
//avoid the special danger that patents applied to a free program could
//make it effectively proprietary.  To prevent this, the GPL assures that
//patents cannot be used to render the program non-free.
```

```
//  The precise terms and conditions for copying, distribution and
//modification follow.

//                       TERMS AND CONDITIONS

//  0. Definitions.

//  "This License" refers to version 3 of the GNU General Public License.

//  "Copyright" also means copyright-like laws that apply to other kinds of
//works, such as semiconductor masks.

//  "The Program" refers to any copyrightable work licensed under this
//License.  Each licensee is addressed as "you".  "Licensees" and
//"recipients" may be individuals or organizations.

//  To "modify" a work means to copy from or adapt all or part of the work
//in a fashion requiring copyright permission, other than the making of an
//exact copy.  The resulting work is called a "modified version" of the
//earlier work or a work "based on" the earlier work.

//  A "covered work" means either the unmodified Program or a work based
//on the Program.

//  To "propagate" a work means to do anything with it that, without
//permission, would make you directly or secondarily liable for
//infringement under applicable copyright law, except executing it on a
//computer or modifying a private copy.  Propagation includes copying,
//distribution (with or without modification), making available to the
//public, and in some countries other activities as well.

//  To "convey" a work means any kind of propagation that enables other
//parties to make or receive copies.  Mere interaction with a user through
//a computer network, with no transfer of a copy, is not conveying.

//  An interactive user interface displays "Appropriate Legal Notices"
//to the extent that it includes a convenient and prominently visible
//feature that (1) displays an appropriate copyright notice, and (2)
//tells the user that there is no warranty for the work (except to the
//extent that warranties are provided), that licensees may convey the
//work under this License, and how to view a copy of this License.  If
//the interface presents a list of user commands or options, such as a
//menu, a prominent item in the list meets this criterion.

//  1. Source Code.

//  The "source code" for a work means the preferred form of the work
//for making modifications to it.  "Object code" means any non-source
//form of a work.

//  A "Standard Interface" means an interface that either is an official
//standard defined by a recognized standards body, or, in the case of
//interfaces specified for a particular programming language, one that
//is widely used among developers working in that language.

//  The "System Libraries" of an executable work include anything, other
//than the work as a whole, that (a) is included in the normal form of
//packaging a Major Component, but which is not part of that Major
//Component, and (b) serves only to enable use of the work with that
//Major Component, or to implement a Standard Interface for which an
//implementation is available to the public in source code form.  A
//"Major Component", in this context, means a major essential component
//(kernel, window system, and so on) of the specific operating system
//(if any) on which the executable work runs, or a compiler used to
//produce the work, or an object code interpreter used to run it.

//  The "Corresponding Source" for a work in object code form means all
//the source code needed to generate, install, and (for an executable
//work) run the object code and to modify the work, including scripts to
```

```
//control those activities.  However, it does not include the work's
//System Libraries, or general-purpose tools or generally available free
//programs which are used unmodified in performing those activities but
//which are not part of the work.  For example, Corresponding Source
//includes interface definition files associated with source files for
//the work, and the source code for shared libraries and dynamically
//linked subprograms that the work is specifically designed to require,
//such as by intimate data communication or control flow between those
//subprograms and other parts of the work.

//  The Corresponding Source need not include anything that users
//can regenerate automatically from other parts of the Corresponding
//Source.

//  The Corresponding Source for a work in source code form is that
//same work.

//  2. Basic Permissions.

//  All rights granted under this License are granted for the term of
//copyright on the Program, and are irrevocable provided the stated
//conditions are met.  This License explicitly affirms your unlimited
//permission to run the unmodified Program.  The output from running a
//covered work is covered by this License only if the output, given its
//content, constitutes a covered work.  This License acknowledges your
//rights of fair use or other equivalent, as provided by copyright law.

//  You may make, run and propagate covered works that you do not
//convey, without conditions so long as your license otherwise remains
//in force.  You may convey covered works to others for the sole purpose
//of having them make modifications exclusively for you, or provide you
//with facilities for running those works, provided that you comply with
//the terms of this License in conveying all material for which you do
//not control copyright.  Those thus making or running the covered works
//for you must do so exclusively on your behalf, under your direction
//and control, on terms that prohibit them from making any copies of
//your copyrighted material outside their relationship with you.

//  Conveying under any other circumstances is permitted solely under
//the conditions stated below.  Sublicensing is not allowed; section 10
//makes it unnecessary.

//  3. Protecting Users' Legal Rights From Anti-Circumvention Law.

//  No covered work shall be deemed part of an effective technological
//measure under any applicable law fulfilling obligations under article
//11 of the WIPO copyright treaty adopted on 20 December 1996, or
//similar laws prohibiting or restricting circumvention of such
//measures.

//  When you convey a covered work, you waive any legal power to forbid
//circumvention of technological measures to the extent such circumvention
//is effected by exercising rights under this License with respect to
//the covered work, and you disclaim any intention to limit operation or
//modification of the work as a means of enforcing, against the work's
//users, your or third parties' legal rights to forbid circumvention of
//technological measures.

//  4. Conveying Verbatim Copies.

//  You may convey verbatim copies of the Program's source code as you
//receive it, in any medium, provided that you conspicuously and
//appropriately publish on each copy an appropriate copyright notice;
//keep intact all notices stating that this License and any
//non-permissive terms added in accord with section 7 apply to the code;
//keep intact all notices of the absence of any warranty; and give all
//recipients a copy of this License along with the Program.

//  You may charge any price or no price for each copy that you convey,
```

```
//and you may offer support or warranty protection for a fee.

//  5. Conveying Modified Source Versions.

//  You may convey a work based on the Program, or the modifications to
//produce it from the Program, in the form of source code under the
//terms of section 4, provided that you also meet all of these conditions:

//    a) The work must carry prominent notices stating that you modified
//    it, and giving a relevant date.

//    b) The work must carry prominent notices stating that it is
//    released under this License and any conditions added under section
//    7.  This requirement modifies the requirement in section 4 to
//    "keep intact all notices".

//    c) You must license the entire work, as a whole, under this
//    License to anyone who comes into possession of a copy.  This
//    License will therefore apply, along with any applicable section 7
//    additional terms, to the whole of the work, and all its parts,
//    regardless of how they are packaged.  This License gives no
//    permission to license the work in any other way, but it does not
//    invalidate such permission if you have separately received it.

//    d) If the work has interactive user interfaces, each must display
//    Appropriate Legal Notices; however, if the Program has interactive
//    interfaces that do not display Appropriate Legal Notices, your
//    work need not make them do so.

//  A compilation of a covered work with other separate and independent
//works, which are not by their nature extensions of the covered work,
//and which are not combined with it such as to form a larger program,
//in or on a volume of a storage or distribution medium, is called an
//"aggregate" if the compilation and its resulting copyright are not
//used to limit the access or legal rights of the compilation's users
//beyond what the individual works permit.  Inclusion of a covered work
//in an aggregate does not cause this License to apply to the other
//parts of the aggregate.

//  6. Conveying Non-Source Forms.

//  You may convey a covered work in object code form under the terms
//of sections 4 and 5, provided that you also convey the
//machine-readable Corresponding Source under the terms of this License,
//in one of these ways:

//    a) Convey the object code in, or embodied in, a physical product
//    (including a physical distribution medium), accompanied by the
//    Corresponding Source fixed on a durable physical medium
//    customarily used for software interchange.

//    b) Convey the object code in, or embodied in, a physical product
//    (including a physical distribution medium), accompanied by a
//    written offer, valid for at least three years and valid for as
//    long as you offer spare parts or customer support for that product
//    model, to give anyone who possesses the object code either (1) a
//    copy of the Corresponding Source for all the software in the
//    product that is covered by this License, on a durable physical
//    medium customarily used for software interchange, for a price no
//    more than your reasonable cost of physically performing this
//    conveying of source, or (2) access to copy the
//    Corresponding Source from a network server at no charge.

//    c) Convey individual copies of the object code with a copy of the
//    written offer to provide the Corresponding Source.  This
//    alternative is allowed only occasionally and noncommercially, and
//    only if you received the object code with such an offer, in accord
//    with subsection 6b.
```

```
//    d) Convey the object code by offering access from a designated
//    place (gratis or for a charge), and offer equivalent access to the
//    Corresponding Source in the same way through the same place at no
//    further charge.  You need not require recipients to copy the
//    Corresponding Source along with the object code.  If the place to
//    copy the object code is a network server, the Corresponding Source
//    may be on a different server (operated by you or a third party)
//    that supports equivalent copying facilities, provided you maintain
//    clear directions next to the object code saying where to find the
//    Corresponding Source.  Regardless of what server hosts the
//    Corresponding Source, you remain obligated to ensure that it is
//    available for as long as needed to satisfy these requirements.

//    e) Convey the object code using peer-to-peer transmission, provided
//    you inform other peers where the object code and Corresponding
//    Source of the work are being offered to the general public at no
//    charge under subsection 6d.

//  A separable portion of the object code, whose source code is excluded
//from the Corresponding Source as a System Library, need not be
//included in conveying the object code work.

//  A "User Product" is either (1) a "consumer product", which means any
//tangible personal property which is normally used for personal, family,
//or household purposes, or (2) anything designed or sold for incorporation
//into a dwelling.  In determining whether a product is a consumer product,
//doubtful cases shall be resolved in favor of coverage.  For a particular
//product received by a particular user, "normally used" refers to a
//typical or common use of that class of product, regardless of the status
//of the particular user or of the way in which the particular user
//actually uses, or expects or is expected to use, the product.  A product
//is a consumer product regardless of whether the product has substantial
//commercial, industrial or non-consumer uses, unless such uses represent
//the only significant mode of use of the product.

//  "Installation Information" for a User Product means any methods,
//procedures, authorization keys, or other information required to install
//and execute modified versions of a covered work in that User Product from
//a modified version of its Corresponding Source.  The information must
//suffice to ensure that the continued functioning of the modified object
//code is in no case prevented or interfered with solely because
//modification has been made.

//  If you convey an object code work under this section in, or with, or
//specifically for use in, a User Product, and the conveying occurs as
//part of a transaction in which the right of possession and use of the
//User Product is transferred to the recipient in perpetuity or for a
//fixed term (regardless of how the transaction is characterized), the
//Corresponding Source conveyed under this section must be accompanied
//by the Installation Information.  But this requirement does not apply
//if neither you nor any third party retains the ability to install
//modified object code on the User Product (for example, the work has
//been installed in ROM).

//  The requirement to provide Installation Information does not include a
//requirement to continue to provide support service, warranty, or updates
//for a work that has been modified or installed by the recipient, or for
//the User Product in which it has been modified or installed.  Access to a
//network may be denied when the modification itself materially and
//adversely affects the operation of the network or violates the rules and
//protocols for communication across the network.

//  Corresponding Source conveyed, and Installation Information provided,
//in accord with this section must be in a format that is publicly
//documented (and with an implementation available to the public in
//source code form), and must require no special password or key for
//unpacking, reading or copying.

//  7. Additional Terms.
```

```
//  "Additional permissions" are terms that supplement the terms of this
//License by making exceptions from one or more of its conditions.
//Additional permissions that are applicable to the entire Program shall
//be treated as though they were included in this License, to the extent
//that they are valid under applicable law.  If additional permissions
//apply only to part of the Program, that part may be used separately
//under those permissions, but the entire Program remains governed by
//this License without regard to the additional permissions.

//  When you convey a copy of a covered work, you may at your option
//remove any additional permissions from that copy, or from any part of
//it.  (Additional permissions may be written to require their own
//removal in certain cases when you modify the work.)  You may place
//additional permissions on material, added by you to a covered work,
//for which you have or can give appropriate copyright permission.

//  Notwithstanding any other provision of this License, for material you
//add to a covered work, you may (if authorized by the copyright holders of
//that material) supplement the terms of this License with terms:

//    a) Disclaiming warranty or limiting liability differently from the
//    terms of sections 15 and 16 of this License; or

//    b) Requiring preservation of specified reasonable legal notices or
//    author attributions in that material or in the Appropriate Legal
//    Notices displayed by works containing it; or

//    c) Prohibiting misrepresentation of the origin of that material, or
//    requiring that modified versions of such material be marked in
//    reasonable ways as different from the original version; or

//    d) Limiting the use for publicity purposes of names of licensors or
//    authors of the material; or

//    e) Declining to grant rights under trademark law for use of some
//    trade names, trademarks, or service marks; or

//    f) Requiring indemnification of licensors and authors of that
//    material by anyone who conveys the material (or modified versions of
//    it) with contractual assumptions of liability to the recipient, for
//    any liability that these contractual assumptions directly impose on
//    those licensors and authors.

//  All other non-permissive additional terms are considered "further
//restrictions" within the meaning of section 10.  If the Program as you
//received it, or any part of it, contains a notice stating that it is
//governed by this License along with a term that is a further
//restriction, you may remove that term.  If a license document contains
//a further restriction but permits relicensing or conveying under this
//License, you may add to a covered work material governed by the terms
//of that license document, provided that the further restriction does
//not survive such relicensing or conveying.

//  If you add terms to a covered work in accord with this section, you
//must place, in the relevant source files, a statement of the
//additional terms that apply to those files, or a notice indicating
//where to find the applicable terms.

//  Additional terms, permissive or non-permissive, may be stated in the
//form of a separately written license, or stated as exceptions;
//the above requirements apply either way.

//  8. Termination.

//  You may not propagate or modify a covered work except as expressly
//provided under this License.  Any attempt otherwise to propagate or
//modify it is void, and will automatically terminate your rights under
//this License (including any patent licenses granted under the third
```

```
//paragraph of section 11).

//  However, if you cease all violation of this License, then your
//license from a particular copyright holder is reinstated (a)
//provisionally, unless and until the copyright holder explicitly and
//finally terminates your license, and (b) permanently, if the copyright
//holder fails to notify you of the violation by some reasonable means
//prior to 60 days after the cessation.

//  Moreover, your license from a particular copyright holder is
//reinstated permanently if the copyright holder notifies you of the
//violation by some reasonable means, this is the first time you have
//received notice of violation of this License (for any work) from that
//copyright holder, and you cure the violation prior to 30 days after
//your receipt of the notice.

//  Termination of your rights under this section does not terminate the
//licenses of parties who have received copies or rights from you under
//this License.  If your rights have been terminated and not permanently
//reinstated, you do not qualify to receive new licenses for the same
//material under section 10.

//  9. Acceptance Not Required for Having Copies.

//  You are not required to accept this License in order to receive or
//run a copy of the Program.  Ancillary propagation of a covered work
//occurring solely as a consequence of using peer-to-peer transmission
//to receive a copy likewise does not require acceptance.  However,
//nothing other than this License grants you permission to propagate or
//modify any covered work.  These actions infringe copyright if you do
//not accept this License.  Therefore, by modifying or propagating a
//covered work, you indicate your acceptance of this License to do so.

//  10. Automatic Licensing of Downstream Recipients.

//  Each time you convey a covered work, the recipient automatically
//receives a license from the original licensors, to run, modify and
//propagate that work, subject to this License.  You are not responsible
//for enforcing compliance by third parties with this License.

//  An "entity transaction" is a transaction transferring control of an
//organization, or substantially all assets of one, or subdividing an
//organization, or merging organizations.  If propagation of a covered
//work results from an entity transaction, each party to that
//transaction who receives a copy of the work also receives whatever
//licenses to the work the party's predecessor in interest had or could
//give under the previous paragraph, plus a right to possession of the
//Corresponding Source of the work from the predecessor in interest, if
//the predecessor has it or can get it with reasonable efforts.

//  You may not impose any further restrictions on the exercise of the
//rights granted or affirmed under this License.  For example, you may
//not impose a license fee, royalty, or other charge for exercise of
//rights granted under this License, and you may not initiate litigation
//(including a cross-claim or counterclaim in a lawsuit) alleging that
//any patent claim is infringed by making, using, selling, offering for
//sale, or importing the Program or any portion of it.

//  11. Patents.

//  A "contributor" is a copyright holder who authorizes use under this
//License of the Program or a work on which the Program is based.  The
//work thus licensed is called the contributor's "contributor version".

//  A contributor's "essential patent claims" are all patent claims
//owned or controlled by the contributor, whether already acquired or
//hereafter acquired, that would be infringed by some manner, permitted
//by this License, of making, using, or selling its contributor version,
//but do not include claims that would be infringed only as a
```

```
//consequence of further modification of the contributor version.  For
//purposes of this definition, "control" includes the right to grant
//patent sublicenses in a manner consistent with the requirements of
//this License.

//  Each contributor grants you a non-exclusive, worldwide, royalty-free
//patent license under the contributor's essential patent claims, to
//make, use, sell, offer for sale, import and otherwise run, modify and
//propagate the contents of its contributor version.

//  In the following three paragraphs, a "patent license" is any express
//agreement or commitment, however denominated, not to enforce a patent
//(such as an express permission to practice a patent or covenant not to
//sue for patent infringement).  To "grant" such a patent license to a
//party means to make such an agreement or commitment not to enforce a
//patent against the party.

//  If you convey a covered work, knowingly relying on a patent license,
//and the Corresponding Source of the work is not available for anyone
//to copy, free of charge and under the terms of this License, through a
//publicly available network server or other readily accessible means,
//then you must either (1) cause the Corresponding Source to be so
//available, or (2) arrange to deprive yourself of the benefit of the
//patent license for this particular work, or (3) arrange, in a manner
//consistent with the requirements of this License, to extend the patent
//license to downstream recipients.  "Knowingly relying" means you have
//actual knowledge that, but for the patent license, your conveying the
//covered work in a country, or your recipient's use of the covered work
//in a country, would infringe one or more identifiable patents in that
//country that you have reason to believe are valid.

//  If, pursuant to or in connection with a single transaction or
//arrangement, you convey, or propagate by procuring conveyance of, a
//covered work, and grant a patent license to some of the parties
//receiving the covered work authorizing them to use, propagate, modify
//or convey a specific copy of the covered work, then the patent license
//you grant is automatically extended to all recipients of the covered
//work and works based on it.

//  A patent license is "discriminatory" if it does not include within
//the scope of its coverage, prohibits the exercise of, or is
//conditioned on the non-exercise of one or more of the rights that are
//specifically granted under this License.  You may not convey a covered
//work if you are a party to an arrangement with a third party that is
//in the business of distributing software, under which you make payment
//to the third party based on the extent of your activity of conveying
//the work, and under which the third party grants, to any of the
//parties who would receive the covered work from you, a discriminatory
//patent license (a) in connection with copies of the covered work
//conveyed by you (or copies made from those copies), or (b) primarily
//for and in connection with specific products or compilations that
//contain the covered work, unless you entered into that arrangement,
//or that patent license was granted, prior to 28 March 2007.

//  Nothing in this License shall be construed as excluding or limiting
//any implied license or other defenses to infringement that may
//otherwise be available to you under applicable patent law.

//  12. No Surrender of Others' Freedom.

//  If conditions are imposed on you (whether by court order, agreement or
//otherwise) that contradict the conditions of this License, they do not
//excuse you from the conditions of this License.  If you cannot convey a
//covered work so as to satisfy simultaneously your obligations under this
//License and any other pertinent obligations, then as a consequence you may
//not convey it at all.  For example, if you agree to terms that obligate you
//to collect a royalty for further conveying from those to whom you convey
//the Program, the only way you could satisfy both those terms and this
//License would be to refrain entirely from conveying the Program.
```

```
//  13. Use with the GNU Affero General Public License.

//  Notwithstanding any other provision of this License, you have
//permission to link or combine any covered work with a work licensed
//under version 3 of the GNU Affero General Public License into a single
//combined work, and to convey the resulting work.  The terms of this
//License will continue to apply to the part which is the covered work,
//but the special requirements of the GNU Affero General Public License,
//section 13, concerning interaction through a network will apply to the
//combination as such.

//  14. Revised Versions of this License.

//  The Free Software Foundation may publish revised and/or new versions of
//the GNU General Public License from time to time.  Such new versions will
//be similar in spirit to the present version, but may differ in detail to
//address new problems or concerns.

//  Each version is given a distinguishing version number.  If the
//Program specifies that a certain numbered version of the GNU General
//Public License "or any later version" applies to it, you have the
//option of following the terms and conditions either of that numbered
//version or of any later version published by the Free Software
//Foundation.  If the Program does not specify a version number of the
//GNU General Public License, you may choose any version ever published
//by the Free Software Foundation.

//  If the Program specifies that a proxy can decide which future
//versions of the GNU General Public License can be used, that proxy's
//public statement of acceptance of a version permanently authorizes you
//to choose that version for the Program.

//  Later license versions may give you additional or different
//permissions.  However, no additional obligations are imposed on any
//author or copyright holder as a result of your choosing to follow a
//later version.

//  15. Disclaimer of Warranty.

//  THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
//APPLICABLE LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT
//HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY
//OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
//THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
//PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM
//IS WITH YOU.  SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF
//ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

//  16. Limitation of Liability.

//  IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
//WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS
//THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
//GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
//USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF
//DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
//PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
//EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
//SUCH DAMAGES.

//  17. Interpretation of Sections 15 and 16.

//  If the disclaimer of warranty and limitation of liability provided
//above cannot be given local legal effect according to their terms,
//reviewing courts shall apply local law that most closely approximates
//an absolute waiver of all civil liability in connection with the
//Program, unless a warranty or assumption of liability accompanies a
//copy of the Program in return for a fee.
```

```
//                     END OF TERMS AND CONDITIONS

//            How to Apply These Terms to Your New Programs

//  If you develop a new program, and you want it to be of the greatest
//possible use to the public, the best way to achieve this is to make it
//free software which everyone can redistribute and change under these terms.

//  To do so, attach the following notices to the program.  It is safest
//to attach them to the start of each source file to most effectively
//state the exclusion of warranty; and each file should have at least
//the "copyright" line and a pointer to where the full notice is found.

//    <one line to give the program's name and a brief idea of what it does.>
//    Copyright (C) <year>  <name of author>

//    This program is free software: you can redistribute it and/or modify
//    it under the terms of the GNU General Public License as published by
//    the Free Software Foundation, either version 3 of the License, or
//    (at your option) any later version.

//    This program is distributed in the hope that it will be useful,
//    but WITHOUT ANY WARRANTY; without even the implied warranty of
//    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
//    GNU General Public License for more details.

//    You should have received a copy of the GNU General Public License
//    along with this program.  If not, see <http://www.gnu.org/licenses/>.

//Also add information on how to contact you by electronic and paper mail.

//  If the program does terminal interaction, make it output a short
//notice like this when it starts in an interactive mode:

//    <program>  Copyright (C) <year>  <name of author>
//    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
//    This is free software, and you are welcome to redistribute it
//    under certain conditions; type `show c' for details.
//

//The hypothetical commands `show w' and `show c' should show the appropriate
//parts of the General Public License.  Of course, your program's commands
//might be different; for a GUI interface, you would use an "about box".

//  You should also get your employer (if you work as a programmer) or school,
//if any, to sign a "copyright disclaimer" for the program, if necessary.
//For more information on this, and how to apply and follow the GNU GPL, see
//<http://www.gnu.org/licenses/>.

//  The GNU General Public License does not permit incorporating your program
//into proprietary programs.  If your program is a subroutine library, you
//may consider it more useful to permit linking proprietary applications with
//the library.  If this is what you want to do, use the GNU Lesser General
//Public License instead of this License.  But first, please read
//<http://www.gnu.org/philosophy/why-not-lgpl.html>.

//Model description
//MEGA is implemented on a 2D cellular grid, and cells can have three possible states: solid, gas, or
//water. The pore structure in a porous medium is represented by a series of shelves that represent
//solids, where the quantity and length of shelves may be set according to the physical characteristics of
//the porous medium being considered. Within MEGA, bubbles may enter the shelf arrangement at a constant
//volumetric rate representing bubble formation via microbial activity or via gas coming out of solution.
//The movement of bubbles is governed by a set of rules that are executed as the shelf arrangement is
//scanned from top to the bottom of the cellular grid (Figure 1).
```

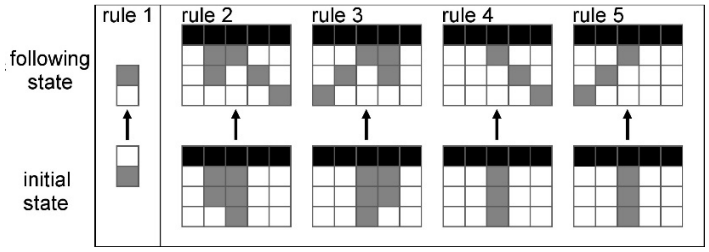

Figure 1. Bubble movement and toppling rules (gas = grey, solid = black, and water = white)

```
//During a simulation, the upward movement a single bubble is replicated by calling the bubble movement
//rule every model iteration until a bubble encounters a shelf (Figure 1, rule 1). If a shelf is vacant,
//the upward movement of the bubble stops. If a single bubble encounters a shelf with stored bubbles, the
//single bubble coalesces with the stored bubbles.  This process increases bubble storage underneath a
//shelf and may trigger bubbles to avalanche upwards to become trapped on shallower shelves or to exit
//from the porous medium into overlying water or air. Bubble avalanches are dependent on the height of
//bubble accumulations that topple according to a rule set (Figure 1, rules 2-5). Bubbles 'topple' upwards
//when bubble columns have a height of three cells, and the direction of toppling (left or right) for free
//standing bubble columns (Figure 1, rules 4 and 5) is decided randomly. MEGA is written in C# and can be
//compiled with Microsoft Visual Studio Express.

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Threading;
using System.Xml;
using Troschuetz.Random;


namespace WindowsApplication1
{
    public partial class Form1 : Form
    {
        //Global variables
        //bubble trap variables
        int[] bubbleSize, bubbleSize2;
        double[] bubbleStart, bubbleStart2;
        int[] bubbleGapCounter, bubbleGapCounter2;
        bool[] bubbleFlag, bubbleFlag2;
        int bubbleChainGap;
        int[] allBubbleSizes;
        public static double magnifyValue;
        private Graphics mygraphics;
        string FILENAME;
        StreamWriter sr30;
        int outflux30minTrap = 0;
        bool bubbleAdded = false;
        int estimatedIter = 0;
        StreamReader sr200;
        double simDuration = 0;
        string current = Directory.GetCurrentDirectory();
        DirectoryInfo di = new DirectoryInfo(Directory.GetCurrentDirectory());
        int[,] elev;
        int[] bubbleSignalShallow;
        int[] bubbleSignalBottom;
        int[] bubbleSignalMid;
        int[] outflux30minFunnelTrap;
        int ncols, nrows;
        Random randy = new Random();
        Random randy2 = new Random();
        int decision = 0;
```

```
        int x, y, y2;
        double inputTimeStepIterations = 0;
        int iteration = 0; //completion of one scan (time in seconds it takes a 1 mm diameter to move 1
mm)
        double realTime = 0; //time in seconds
        double bubbleSpeed = 0; //in mm per second, equivalent to number of iterations in 1 second
        int intBubbleSpeed = 0; //bubble speed as integer
        //double bubbleTimeShallow = 0; //in iterations, how many bubbles are produced in x seconds
shallow
        double bubbleTimeBottom = 0; //in iterations, how many bubbles are produced in x seconds at bottom
        //double bubbleTimeMid = 0; //in iterations, how many bubbles are produced in x seconds in middle
        double adjustedBubbleSpeed2; //iteration = 1/282 sec or 1/185sec or 1/6sec
        int iteration1hr = 0;
        int iteration30min = 0;
        int iteration1day = 0;
        int iteration10day = 0;
        int iteration60sec = 0;
        int randomRow = 0;
        int randomCol = 0;
        //interface Variables
        private System.Drawing.Bitmap m_objDrawingSurface;
        int graphics_scale = 2;
        SolidBrush Peatbrush;
        SolidBrush Bubblebrush;
        SolidBrush Waterbrush;
        SolidBrush Guidebrush;
        bool noRules;
        Graphics objGraphics;

        public Form1()
        {
            InitializeComponent();
        }

        //METHODS
        private void initialise()
        {
            mygraphics = this.CreateGraphics();
            bubbleSpeed = System.Convert.ToDouble(bubbleSpeedtextBox20.Text);
            intBubbleSpeed = System.Convert.ToInt32(bubbleSpeed);
            adjustedBubbleSpeed2 = 1.0 / bubbleSpeed;
            simDuration = System.Convert.ToDouble(simDurationTextBox.Text);
            elev = new int[nrows, ncols];
            bubbleSignalShallow = new int[100000];
            outflux30minFunnelTrap = new int[12];
            bubbleSignalBottom = new int[100000];
            bubbleSignalMid = new int[100000];
            estimatedIter = System.Convert.ToInt32(bubbleSpeed * simDuration);
            inputTimeStepIterations = System.Convert.ToDouble(inputSteptextBox20.Text) * bubbleSpeed;
            iteration1hr = System.Convert.ToInt32(bubbleSpeed) * 3600;
            iteration30min = System.Convert.ToInt32(bubbleSpeed) * 1800;
            iteration1day = System.Convert.ToInt32(bubbleSpeed) * 3600 * 24;
            iteration10day = iteration1day * 10;
            iteration60sec = System.Convert.ToInt32(bubbleSpeed) * 60;
            iteration = iteration1day - 10;//start simulation on second day
            //bubble trap variables
            bubbleSize = new int[ncols];
            bubbleSize2 = new int[ncols];
            bubbleStart = new double[ncols];
            bubbleStart2 = new double[ncols];
            bubbleGapCounter = new int[ncols];
            bubbleGapCounter2 = new int[ncols];
            bubbleFlag = new bool[ncols];
            bubbleFlag2 = new bool[ncols];
            bubbleChainGap = System.Convert.ToInt32(sepDistTextBox1.Text);
            allBubbleSizes = new int[10000];
            Peatbrush = new SolidBrush(Color.FromArgb(255, 255, 0, 0));//red
            Bubblebrush = new SolidBrush(Color.FromArgb(255, 255, 255, 255));//white
            Waterbrush = new SolidBrush(Color.FromArgb(255, 0, 0, 0));//black
```

```
            Guidebrush = new SolidBrush(Color.FromArgb(255, 0, 0, 255));//blue
        }
        private void button2_Click(object sender, EventArgs e)//start button
        {
            startButton.Enabled = false;
            backgroundWorker1.RunWorkerAsync();
        }
        //###################################PROGRAM STARTS HERE###################################
        private void main_loop()
        {
            //SCANNING ARRAY and bubble movement and toppling
            do
            {
                if (noRules == false)//keep scanning because bubbles are in motion
                {
                    noRules = true;//assume bubbleas are not in motion
                    for (int xTopple = 1; xTopple <= nrows - 1; xTopple++)
                    {
                        for (int yTopple = 1; yTopple <= ncols - 1; yTopple++)
                        {
                            ////Exit top 1 white bubbles and record flux
                            if ((xTopple == 1) && (elev[xTopple, yTopple] == 1))
                            {
                                elev[xTopple, yTopple] = 0;
                                noRules = false;//bubbles in motion
                            }
                            //Rule 1 white bubbles
                            else if ((elev[xTopple, yTopple] == 1) && (elev[xTopple - 1, yTopple] == 0))
                            {
                                elev[xTopple - 1, yTopple] = 1;
                                elev[xTopple, yTopple] = 0;
                                noRules = false;//bubbles in motion
                            }
                            ////Rule2, freestanding
                            else if ((elev[xTopple, yTopple] == 1) && (elev[xTopple - 1, yTopple] == 1) &&
(elev[xTopple - 2, yTopple] == 1)
                                && (elev[xTopple, yTopple - 1] == 0) && (elev[xTopple - 1, yTopple - 1] ==
0) && (elev[xTopple - 2, yTopple - 1] == 0)
                                && (elev[xTopple, yTopple - 2] == 0) && (elev[xTopple, yTopple + 1] == 0)
&& (elev[xTopple - 1, yTopple + 1] == 0) &&
                                (elev[xTopple - 2, yTopple + 1] == 0) && (elev[xTopple, yTopple + 2] ==
0))
                            {
                                decision = randy.Next(0, 2);
                                if (decision == 0) //topple left
                                {
                                    elev[xTopple, yTopple] = 0;
                                    elev[xTopple - 1, yTopple] = 0;
                                    elev[xTopple - 1, yTopple - 1] = 1;
                                    elev[xTopple, yTopple - 2] = 1;
                                    noRules = false;//bubbles in motion
                                }
                                else
                                {
                                    elev[xTopple, yTopple] = 0;
                                    elev[xTopple - 1, yTopple] = 0;
                                    elev[xTopple - 1, yTopple + 1] = 1;
                                    elev[xTopple, yTopple + 2] = 1;
                                    noRules = false;//bubbles in motion
                                }
                            }
                            //Rule3, topple left
                            else if ((elev[xTopple, yTopple] == 1) && (elev[xTopple - 1, yTopple] == 1) &&
(elev[xTopple - 2, yTopple] == 1)
                                && (elev[xTopple, yTopple - 1] == 0) && (elev[xTopple - 1, yTopple - 1] ==
0) && (elev[xTopple - 2, yTopple - 1] == 0) &&
                                (elev[xTopple, yTopple - 2] == 0))
                            {
                                elev[xTopple, yTopple] = 0;
```

```
                                elev[xTopple - 1, yTopple] = 0;
                                elev[xTopple - 1, yTopple - 1] = 1;
                                elev[xTopple, yTopple - 2] = 1;
                                noRules = false;//bubbles in motion
                            }
                            //Rule4, topple right
                            else if ((elev[xTopple, yTopple] == 1) && (elev[xTopple - 1, yTopple] == 1) &&
(elev[xTopple - 2, yTopple] == 1)
                                && (elev[xTopple, yTopple + 1] == 0) && (elev[xTopple - 1, yTopple + 1] ==
0) && (elev[xTopple - 2, yTopple + 1] == 0)
                                && (elev[xTopple, yTopple + 2] == 0))
                            {
                                elev[xTopple, yTopple] = 0;
                                elev[xTopple - 1, yTopple] = 0;
                                elev[xTopple - 1, yTopple + 1] = 1;
                                elev[xTopple, yTopple + 2] = 1;
                                noRules = false;//bubbles in motion
                            }
                        }
                    }
                }
                //measure bubble size and flux
                if (noRules == false)//last scan a rule was called
                {
                    //global bubble trap
                    for (y = 100; y <= 1100; y++)//scan across small bubble trap
                    {
                        x = 5;//bubble trap on 5th row, from top
                        if (y % 2 == 0) //bubble trap
                        {
                            //check if bubble exists in trap
                            if ((elev[x, y] == 1) || (elev[x, y + 1] == 1))//bubble exists
                            {
                                //record the bubble
                                if (elev[x, y] == 1)
                                {
                                    bubbleSize[y]++;
                                    if (bubbleGapCounter[y] > 0)//gap was previously found
                                    {
                                        bubbleGapCounter[y] = 0;//resets gap counter if new bubble found
                                    }
                                    bubbleFlag[y] = true;//open chain
                                }
                                if (elev[x, y + 1] == 1)
                                {
                                    bubbleSize[y]++;
                                    if (bubbleGapCounter[y] > 0)//gap was previously found
                                    {
                                        bubbleGapCounter[y] = 0;//resets gap counter if new bubble found
                                    }
                                    bubbleFlag[y] = true;//open chain
                                }
                            }
                            if ((elev[x, y] == 0) && (elev[x, y + 1] == 0))//a gap
                            {
                                if (bubbleFlag[y] == true)
                                {
                                    bubbleGapCounter[y]++;//keep track of gaps
                                    if (bubbleGapCounter[y] >= bubbleChainGap)//chain is over, close chain
(3mm gap)
                                    {
                                        allBubbleSizes[bubbleSize[y]]++;//records bubble size to array
                                        outflux30minTrap = outflux30minTrap + bubbleSize[y];//records
bubble flux
                                        bubbleSize[y] = 0;
                                        bubbleGapCounter[y] = 0;
                                        bubbleStart[y] = 0;
                                        bubbleFlag[y] = false;
                                    }
```

```
                            }
                        }
                    }
                }
                //funnel traps
                for (y = 100; y <= 900; y = y + 200)//scan across small bubble trap
                {
                    for (y2 = y; y2 < y + 200; y2++)//scan acroos 5 funnel traps of 200 mm width
                    {
                        x = 5;//bubble trap on 5th row, from top
                        if (y2 % 2 == 0) //bubble trap
                        {
                            //check if bubble exists in trap
                            if ((elev[x, y2] == 1) || (elev[x, y2 + 1] == 1))//bubble exists
                            {
                                //record the bubble
                                if (elev[x, y2] == 1)
                                {
                                    bubbleSize2[y2]++;
                                    if (bubbleGapCounter2[y2] > 0)//gap was previously found
                                    {
                                        bubbleGapCounter2[y2] = 0;//resets gap counter if new bubble
found
                                    }
                                    bubbleFlag2[y2] = true;//open chain
                                }
                                if (elev[x, y2 + 1] == 1)
                                {
                                    bubbleSize2[y2]++;
                                    if (bubbleGapCounter2[y2] > 0)//gap was previously found
                                    {
                                        bubbleGapCounter2[y2] = 0;//resets gap counter if new bubble
found
                                    }
                                    bubbleFlag2[y2] = true;//open chain
                                }
                            }
                            if ((elev[x, y2] == 0) && (elev[x, y2 + 1] == 0))//a gap
                            {
                                if (bubbleFlag2[y2] == true)
                                {
                                    bubbleGapCounter2[y2]++;//keep track of gaps
                                    if (bubbleGapCounter2[y2] >= bubbleChainGap)//chain is over, close
chain (3mm gap)
                                    {
                                        outflux30minFunnelTrap[y / 100] = outflux30minFunnelTrap[y /
100] + bubbleSize2[y2];//records bubble flux
                                        bubbleSize2[y2] = 0;
                                        bubbleGapCounter2[y2] = 0;
                                        bubbleStart2[y2] = 0;
                                        bubbleFlag2[y2] = false;
                                    }
                                }
                            }
                        }
                    }
                }
                //UPDATE TIME
                iteration++;
                realTime = iteration * adjustedBubbleSpeed2; //iteration =  1/bubblespeed

                //OUTPUTS
                //hourly flux output
                if ((iteration % iteration30min == 0))//every 30 min flux to file
                {
                    sr30 = File.AppendText("Flux30min.csv");
```

```
                    sr30.WriteLine(System.Convert.ToInt32(realTime) + "," + outflux30minTrap + "," +
outflux30minFunnelTrap[1] + "," + outflux30minFunnelTrap[3] + "," + outflux30minFunnelTrap[5] + "," +
outflux30minFunnelTrap[7] + "," + outflux30minFunnelTrap[9]);
                    outflux30minTrap = 0;//reset bubble flux per 30 min
                    outflux30minFunnelTrap[1] = 0;
                    outflux30minFunnelTrap[3] = 0;
                    outflux30minFunnelTrap[5] = 0;
                    outflux30minFunnelTrap[7] = 0;
                    outflux30minFunnelTrap[9] = 0;
                    sr30.Close();
                }
                //every 1 day output of storage and bubble sizes
                if (iteration % iteration1day == 0)//record storage to array every 1 day, and bubble sizes
to file                 {
                {
                    //storage as ascii
                    //save final storage array to file
                    using (StreamWriter sw = new StreamWriter("storage" + System.Convert.ToInt32(realTime)
+ ".asc"))
                    {
                        sw.WriteLine("ncols," + ncols.ToString());
                        sw.WriteLine("nrows," + nrows.ToString());
                        for (x = 0; x < nrows; x++)
                        {
                            for (y = 0; y < ncols; y++)
                            {
                                sw.Write(elev[x, y]);
                                sw.Write(" ");
                            }
                            sw.Write("\n");
                        }
                        sw.Close();
                    }
                    ////bubble sizes
                    FILENAME = "sizes" + System.Convert.ToInt32(realTime) + ".csv";
                    using (StreamWriter sw = new StreamWriter(FILENAME))
                    {
                        for (x = 0; x < 10000; x++)
                        {
                            if (allBubbleSizes[x] != 0)//bubbles must be larger than 0
                            {
                                sw.WriteLine(x + "," + allBubbleSizes[x]);
                            }
                            allBubbleSizes[x] = 0;
                        }
                        sw.Close();
                    }
                }
                //Bubble INPUTS
                if (iteration % inputTimeStepIterations == 0)//x seconds has elapsed, update production
                {
                    bubbleTimeBottom = Math.Round((inputTimeStepIterations /
bubbleSignalBottom[System.Convert.ToInt32(iteration / inputTimeStepIterations)])); //interval in
iterations that a bubble should be added
                }
                if (iteration % bubbleTimeBottom == 0)
                {
                    bubbleAdded = false;
                    while (bubbleAdded == false)
                    {
                        //random locations hardcoded
                        randomCol = randy.Next(100, 1100);//exclude 100 mm on each side of peat
                        randomRow = randy.Next(20, 735);//exclude top 10 mm of peat, and last 10 rows
                        if (elev[randomRow, randomCol] == 0)//open space for bubble
                        {
                            elev[randomRow, randomCol] = 1;
                            bubbleAdded = true;
                        }
                    }
```

```
            noRules = false;//bubbles in motion
        }
    } while (iteration < estimatedIter);
    endSim();//simulation done
}

private void loadData()
{
    //read in peat file header to get ncols, nrows
    string FILE_NAME = this.loaddtextBox.Text;
    StreamReader sr = File.OpenText(FILE_NAME);
    string[] lineArray2;
    lineArray2 = sr.ReadLine().Split(new char[] { ',' });//read ncols from header
    ncols = System.Convert.ToInt32(lineArray2[1]);
    lineArray2 = sr.ReadLine().Split(new char[] { ',' });//read nrows from header
    nrows = System.Convert.ToInt32(lineArray2[1]);
    sr.Close();
    initialise();//initialise variables
    //load peat array from file
    int x, y = 1, xcounter;
    String input, input2;
    int tttt = 0;
    y = 0;
    StreamReader sr10 = File.OpenText(FILE_NAME);
    lineArray2 = sr10.ReadLine().Split(new char[] { ',' });//read ncols from header
    ncols = System.Convert.ToInt32(lineArray2[1]);
    lineArray2 = sr10.ReadLine().Split(new char[] { ',' });//read nrows from header
    nrows = System.Convert.ToInt32(lineArray2[1]);
    while ((input = sr10.ReadLine()) != null)
    {
        string[] lineArray;
        lineArray = input.Split(new char[] { ' ' });
        xcounter = 1;
        for (x = 1; x <= (lineArray.Length - 1); x++)
        {
            if (lineArray[x] != "" && xcounter <= ncols)
            {
                tttt = int.Parse(lineArray[x]);
                elev[y, xcounter] = tttt;
                xcounter++;
            }
        }
        y++;
    }
    sr10.Close();
    int counter = 0;
    //read in bubble signal file-bottom
    string FILE_NAME_BUBBLE_SIGNAL = this.bubbleSignalBottomTextBox.Text;
    sr200 = File.OpenText(FILE_NAME_BUBBLE_SIGNAL);
    counter = 0;
    while ((input2 = sr200.ReadLine()) != null)
    {
        bubbleSignalBottom[counter] = System.Convert.ToInt32(input2);
        counter = counter + 1;
    }
    sr200.Close();

    //make drawing surface
    m_objDrawingSurface = new Bitmap((ncols) * graphics_scale,
        (nrows) * graphics_scale, System.Drawing.Imaging.PixelFormat.Format24bppRgb);
    //drawwater();
    loadButton.Enabled = false;
    startButton.Enabled = true;
}
private void loadButton_Click(object sender, EventArgs e)
{
    loadData();
}
```

```
        private void endSim()
        {
            //save final storage array to file
            string FILENAME = "storageFinal.txt";
            using (StreamWriter sw = new StreamWriter(FILENAME))
            {
                sw.WriteLine("ncols," + ncols.ToString());
                sw.WriteLine("nrows," + nrows.ToString());
                for (x = 0; x < nrows; x++)
                {
                    for (y = 0; y < ncols; y++)
                    {
                        sw.Write(elev[x, y]);
                        sw.Write(" ");
                    }
                    sw.Write("\n");
                }
                sw.Close();
            }
        }

        private void quitSaveButton6_Click(object sender, EventArgs e)//quit and save button
        {
            if (realTime < simDuration)//save files mid-simulation
            {
                endSim();
                this.Close();
            }
            else
            {
                this.Close();
            }
        }

        private void backgroundWorker1_DoWork(object sender, DoWorkEventArgs e)
        {
            main_loop();
        }

        delegate void SetTextCallback(string text);//set time box without cross threading

        private void SetText(string text)//set time box without cross threading
        {
            if (this.timeBox.InvokeRequired)
            {
                SetTextCallback d = new SetTextCallback(SetText);
                this.Invoke(d, new object[] { text });
            }
            else
            {
                this.timeBox.Text = text;
            }
        }
    }
}
```