Metodología de la Programación (I. T. Informática) Manual de estilo para escribir código en C[*]

Francisco José Madrid Cuevas
Dpto. de Informática y Análisis Numérico.
Universidad de Córdoba.

17 de octubre de 2007

1 Por qué definir un estilo de programación

Según Kernighan *et al.*[1]:

> ‘‘Escribir un programa es más que tener una buena sintaxis, corregir los errores y volverlo rápido. Los programas son leídos no sólo por computadoras sino también por los programadores. Un programa bien escrito es más fácil de entender y modificar que uno mal escrito. La disciplina de la buena escritura produce código con más probabilidades de ser correcto. Afortunadamente, esta disciplina no es difícil.”

[*] Este texto supone un resumen del primer capítulo del libro “La práctica de la Programación”, B. W. Kernighan y R. Pike. Pearson Educación, 2000.

Según Mena Quintero *et al.*[2]:

> ‘‘GNOME es un proyecto de software libre muy ambicioso, y como tal está compuesto de muchos paquetes que son más o menos independientes los unos de los otros ... los programadores pueden entrar o salir (del proyecto, n.d.t.) en cualquier momento y serán capaces de dedicar distintas cantidades de tiempo al proyecto GNOME ... La mayoría de los voluntarios ocasionales no comienzan ningún gran proyecto por sí mismos ... concluimos que es muy importante para los proyectos existentes hacerlos tan fácil como sea posible para la gente que contribuye en ellos. Una forma de hacer esto es por medio de asegurar que los programas son fáciles de leer, comprender y modificar.

En general toda entidad que se dedique al desarrollo de software cuenta con un guía para la escritura del software, incluyendo en ella por supuesto la guía de estilo a la hora de escribir el código. Ejemplos son:

- `GNU` : “GNU Coding Standards” [4].
- `GNOME` : “GNOME Programming Guidelines” [2].
- `LINUX` “Linux Kernel Coding style” [3]

2 Estilo

> *“Desde hace mucho tiempo se sabe que los mejores escritores a veces desatienden las reglas de la retórica. Sin embargo, cuando lo hacen, normalmente el lector encuentra en la frase algún mérito en compensación, obtenido a costa de la violación. A menos que se tenga la certeza de hacerlo bien, lo mejor será seguir las reglas.”*
>
> William Strunk y E.B. White, *The Elements of Style*

2.1 Nombres

- **Emplee nombres descriptivos para variables globales, y nombres cortos para variables locales**.

Las funciones locales, clases y estructuras también deben tener nombres descriptivos que denoten su papel dentro del programa.

Para variables globales es útil añadir algún comentario a su definición.

Ejemplo para variable global: `lista_alumnos`

Ejemplo para variables locales: `i`, `nalumns`

Estos dos últimos serán nuestro estilos para nombrar en prácticas.

- **Sea consistente.** Dé a las cosas relacionadas nombres que muestren esta relación y marquen las diferencias.

  Ejemplo:

  ```
  ? class UserQueue
  ? {
  ?    int noOfItemsInQ, frontOfTheQueue, queueCapacity;
  ?    public int noOfUsersInQueue() {...};
  ? }
  ```

  La palabra `Queue` aparece como Q, Queue y queue. Además es redundante ya que la clase de por sí se llama Queue. Una mejor versión sería:

  ```
  ? class UserQueue
  ? {
  ?    int nitems, front, capacity;
  ?    public int nusers() {...};
  ? }
  ```

  aún así sigue confundiendo los conceptos “item” con “user”. Se debería cambiar para utilizar uno sólo.

- **Emplee nombre activos para funciones.**

  Por ejemplo: `obtener_fecha()` Este será nuestro estilo para nombrar funciones.

  Para funciones que retornen un tipo booleano, se debería utilizar nombres que indiquen el sentido del valor de retorno. Por ejemplo, en lugar de nombrar `comprobar_octal()` que no indica claramente cuál valor es verdadero y cuál es falso, es mejor nombrar a la función como `es_octal()` que no provoca ambigüeadad.

- **Sea preciso.** Un nombre no sólo identifica, sino que ofrece información al lector. Un nombre confuso puede ocasionar errores difíciles de detectar. Por ejemplo:

  ```
  ? public boolean inTable(Object obj)
  ? {
  ?   int j= this.getIndex(obj);
  ?   return ( j == nTable );
  ? }
  ```

  La función `inTable()` devuelve un valor entre cero y `nTable-1` si encuentra el objeto y `nTable` si no lo encuentra. Así el valor de retorno de `inTable()` no coincide con lo que de su nombre se desprende.

2.2 Expresiones e instrucciones

- **Como regla general**, escriba las expresiones e instrucciones de forma tal que vuelvan su significado lo más transparente posible.

- **Utilice las sangrías para mostrar la estructura**

  Ejemplo incorrecto:

  ```
  ?  for(n=0;n<100;field[n++]='\0');
  ?  *i='\0'; return('\n');
  ```

  Una mejora sería:

  ```
  ?  for(n=0; n<100; field[n++]='\0')
  ?  ;
  ?  *i='\0';
  ?  return('\n');
  ```

  Una mejora más sería colocar la asignación en el cuerpo del bucle for y separar el incremento:

```
?  for(n=0; n<100; n++)
?     field[n]='\0';
?  *i='\0';
?  return '\n';
```

- **Utilice una forma natural para las expresiones**. Escriba las expresiones tal y como las diría en voz alta.

  Ejemplo:

  ```
  ? if (!(bid < actbs) || !(bid >= unblk))
  ?   ...
  ```

  Esta expresando en negativo, se entenderá mejor así:

  ```
    if ((bid >= actbs) || (bid < unblk))
      ...
  ```

- **Emplee los paréntesis para resolver la ambigüedad**. Algunas veces será para mejorar la legibilidad, otras para escribir código correcto.

  Ejemplo: la expresión `if (x&MASK == BITS)` significa realmente

  `if (x & (MASK == BITS))`

  que seguramente no es la intención del programador, así quedará mejor:

  `if ((x & MASK) == BITS)`

  Ejemplo: el siguiente código no requiere paréntesis, pero

  ```
  ? leap_year = y % 4 == 0 && y%100 != 0 ||
                y % 400 == 0;
  ```

  se leerá mucho mejor si lo escribimos de la forma:

  ```
    leap_year = ((y%4 == 0) && (y%100 != 0)) ||
                (y%400 == 0);
  ```

- **Divida las expresiones complejas**. C tiene una sintaxis de operadores muy rica que permite escribir expresiones como:

  ```
  ? *x += (*xp=(2*k < (n-m) ? c[k+1] : d[k--]));
  ```

  Seguro que es más fácil de leer si se escribe de la siguiente forma:

  ```
  if (2*k < (m-n))
    *xp = c[k+1];
  else
    *xp = d[k--];
  *x += *xp;
  ```

2.3 Consistencia y convenciones

La consistencia da como resultado mejores programas, por ello es necesario definir un "estilo" de codificación y escribir un documento donde se especifique (o adherirse a uno ya definido, por ejemplo "Gnu Coding Standards"[4]).

- **Use un estilo consistente para las sangrías y las llaves**. Hay varios estilos Gnu, KR, el que tú mismo te definas... Lo realmente importante es que se aplique de manera consistente en todo el código.

  Para el estilo Gnu la sangría es de dos espacios por nivel de sangrado. Para el estilo Kernel Linux, la sangría es de ocho espacios. Nosotros usaremos sangrias de dos espacios por nivel de anidamiento.

  Respecto a las llaves el estilo K-R coloca la llave que abre el bloque al final de la sentencia cabecera y la que cierra en la misma columna de inicio de la sentencia cabecera:

  ```
  if (       ) {
   ...
  }
  ```

  mientras otro estilo (GNU) las pone las dos en la primera columna de la sentencia cabecera de la forma:

```
if (      )
  {
   ...
  }
```

Sin embargo el estilo K-R hace una excepción con las funciones utilizando el estilo GNU:

```
int
mi_funcion( int parametro1, int parametro2,
            int parametro3, int parametro4 )
{
  ...
  return 0;
}
```

Las funciones se definen en la primera columna, el tipo devuelto se ponen en la primera línea, en la siguiente el identificador y sus parámetros y las llaves se ponden en la primera columna si indentar.

Nosotros en clase utilizaremos el estilo GNU.

- **Rompa líneas mayores de 80 caracteres.** Cuando una línea es demasiado larga y no coge en una pantalla de ochenta columnas, se romperá en trozos significativos más pequeños dispuestos muy a la derecha de la pantalla:

```
if (condition)
  printf("Esto es un ejemplo de "
                     "sentencia muy "
                     "larga de escribir "
                     "%d, %f\n",
                     a, sqrt(b));
else
  ...
```

- **Una función sólo debería tener un punto de salida**. Mejora la legibilidad del código y es menos propenso a errores.

```
int
una_funcion(...)
{
  if (cond1)
    return 1;
  else if (cond2)
    return 2;
  else if (cond3)
    return 3;
  else
    return 4;
}
```

Quedará mejor de la forma, y esta será nuestra convención para las prácticas, como:

```
int
una_funcion(...)
{
  int ret=4;

  if (cond1)
    ret=1;
  else if (cond2)
    ret=2;
  else if (cond3)
    ret=3;

  return ret;
}
```

2.4 Macros de funciones

- **No defina macros de funciones**.

  Existe la tendencia a definir funciones muy cortas como “macros” del preprocesador para que éste las expanda en el código cuando son invocadas y se evite la sobrecarga en la llamada, incluso la propia biblioteca de C las utiliza (por ejemplo `getchar`)

Hoy en día el estándar el ISO C99 incorpora el especificar de función `inline` que propicia la expansión del código en vez de su invocación, pero manteniendo la comprobación estricta de tipos y la precedencia en la resolución de expresiones.

```
? #define isupper(c) ((c) >= 'A' && (c) <= 'Z')
? ...
? while (isupper(c=getchar())) {
? ...
? }
```

Este código está mal porque lee dos caracteres cada vez que se invoca a la macro (se expande dos veces `c=getchar()`). Si embargo si escribimos:

```
inline bool
isupper(char c)
{
  return ((c) >= 'A' && (c) <= 'Z');
}
```

Ya no ocurrirán estos efectos laterales. Como vamos a utilizar un compilador que cumple el estándar C99, nuestra convención en clase de prácticas será utilizar funciones “inline” en lugar de macros.

- **Ponga paréntesis en el cuerpo y argumentos de las macros**.

```
? #define square(c) c * c
? ...
? valor = 1.0 / square(x);
? ...
?
```

seguramente está mal porque se expande a `1.0/x * x`. La solución sería definir la macro como:

```
? #define square(c) ((c) * (c))
```

pero mejor aún es definir una función `inline`.

2.5 Números mágicos

Entendemos por número mágicos, todos los valores constante literal (constantes numéricas, de texto, tamaños de arreglos, posiciones de carácter dentro de un buffer, factores de conversión) que aparecen en un programa.

- **Dele nombre a estas constantes**. No deje los valores literales desperdigados por el código.
- **Defina los valores literales como constantes, no como macros**. El estándar C99 incorpora el especificador `const` para definir dichas constantes. Anteriormente se recorría a macros del preprocesador. Esto tiene el inconveniente de que no se comprueban los tipos convenientemente:

  ```
  ? #define SALUDO ``Hola mundo''
  ```

  es mejor así:

  ```
    const char * saludo=``Hola mundo'';
  ```

  Como nosotros utilizamos un compilar que cumple el estándar C99, vamos a utilizar variables constantes en vez de macros en las prácticas.

2.6 Comentarios

El propósito de los comentarios es ayudar al lector de un programa. No mencionan cosas que el código ya dice directamente, ni lo contradicen, tampoco distraen al lector con elaborados despliegues tipográficos. Los mejores comentarios ayudan al entendimiento de un programa indicando brevemente los detalles sobresalientes, u ofreciendo un punto de vista más amplio sobre los procedimientos.

- **No repita lo que ya es obvio**.

  Ejemplo: `a = a + 1; /*incrementar variable*/`

- **Comente las funciones y los datos globales**.

  Describa el propósito de una función así como de sus argumentos y todo aquello que pueda ser relevante (bodigos de error retornados, excepciones lanzadas, etc.) en la definición de la función.

  Hoy en día existen herramientas que permiten extraer documentación automáticamente de los comentarios para generar el "manual de código" de un proyecto. Si se comenta bien el código estas pueden suponer una herramienta muy poderosa. Ejemplos de estas herramientas son `Javadoc` para lenguaje Java y `doxygen` para lenguaje C/C++ [5].

- **No contradiga el código, aclare, no confunda**.

Referencias

[1] Brian W. Kernighan Rob Pike. *La Práctica de la Programación*. Pearson Educación, 2000.

[2] Miguel de Icaza M. Wlinder F. Mena Quintero. *GNOME Programming Guidelines*. GNOME project. Available as http://developer.gnome.org/doc/guides/programming-guidelines/book1.html.

[3] Linux Kernel Project. *Linux Kernel Coding Style*. Available as file:///usr/src/linux/Documentation/CodingStyle.

[4] R. Stallman. *GNU Coding Standards*. Free Software Foundation, GNU free software Project. Available as http://www.gnu.org/prep/standards/standards.html.

[5] Dimitri van Heesch. *Doxygen Manual*. Doxygen Proyect. Available as http://www.stack.nl/ dimitri/doxygen/manual.html.

3 GNU Free Documentation License

Version 1.2, November 2002

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free"in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of çopyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **"Document"**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as τ**ou"**. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A **"Modified Version"** of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A **"Secondary Section"** is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **"Invariant Sections"** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **Çover Texts"** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **"Transparent"** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent"is called **.Opaque"**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **"Title Page"** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page"means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section **.Entitled XYZ"** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **.Acknowledgements"**, **"Dedications"**, **.Endorsements"**, or **"History"**.) To **"Preserve the Title"** of such a section when you modify the Document means that it remains a section .Entitled XYZ.ªccording to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History"in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History"section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled .Acknowledgements.ºr "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled .Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled .Endorsements.or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled .Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History"in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled .Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled .Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an .aggregate"if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled .Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation 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. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License .or any later version.applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts."line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.