======== Instructions and Install Log for Knuth's Stanford GraphBase ========
For: Mac OS Version 10.13 or later

By: Tim Nicholson (tim.a.nicholson@gmail.com) - Stanford Class of 1971
File: Install GraphBase Mac OS.rtf

The Stanford GraphBase is a set of programs written in the C Language and has documentation on programs that manipulate data structures called graphs. Donald Knuth's 1993 book "The Stanford GraphBase - A Platform for Combinatorial Computing" describes these programs in detail as a means to allow others to rewrite the software and try other methods or to combine graphs with other data structures.
See: The Stanford GraphBase - A Platform for Combinatorial Computing
By Donald E. Knuth, Copyright 1994 by the ACM Press

Knuth's series of books "The Art of Computer Programming" has an extensive amount of algorithms on combinatorial computing in Volume 4. Volume 4A was published in 2011 and two more volumes should follow soon if all goes well. The Stanford GraphBase is a good companion for those students wishing to test various routines that appear in Volume 4A and the ones to follow. Knuth's CWEB software is also useful and its installation is required to implement these programs.

For more information on CWEB See (i.e. click on):
The CWEB System of Structured Documentation
by Donald E. Knuth and Silvio Levy (Reading, Massachusetts: Addison-Wesley)

And a way to write well documented programs:
Literate Programming
by Donald E. Knuth (Stanford, California: Center for the Study of Language and Information, 1992)

Notes:

1) This file: "Install GraphBase Mac OS.rtf" is created and maintained using the Mac OS "TextEdit" program in the Applications directory.
   See: /Applications/Launchpad.app.

   Any other plain text editor will also work. It is also distributed as a PDF file. The text in blue are links to web sites which you can click on below to see what the link refers to. The text in red are commands entered on the Mac Terminal app which can be invoked by following steps 2a and 2b below. Search for "Mac Terminal Commands" for a list of them. Each terminal command has on-line documentation that can be viewed by entering "man <command-name>". You can recall a previously entered command by using the Up-Arrow key and the other Arrow keys and the Delete key to modify and resubmit the command.

2) Before you install the Stanford GraphBase library you must have Knuth's CWEB programs installed. See: http://macappstore.org/cweb/

   a) Run the Terminal app (Its icon is at the bottom) or:

   b) Press Command+Space and type Terminal and press enter/return key to enter commands to Mac OS. Then enter the command (all on the same line - Note: ruby is an Interpreted object-oriented scripting language. Type "man ruby" for usage. Homebrew is a software installation program.):

   c) `ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)" < /dev/null 2> /dev/null`

   d) Once Homebrew is installed then enter the command:

   e) `brew install cweb`

   f) See Step 4) below to optionally install Knuth's TEX typesetting S/W TEX or its variants. They can be used in CWEB and SGB to typeset the output of the CWEAVE program in CWEB.

3) Download the TAR file that contains the source code and Makefile for the Stanford GraphBase library:

   a) On Knuth's GraphBase website at:
   https://www-cs-faculty.stanford.edu/~knuth/sgb.html

b) Find the following text (i.e. Available from . . .) on that webpage and click on "master sources" and a file sgb.tar will be downloaded into your Downloads directory. Move the TAR file to your source directory and then double-click to unpack the archive files into that source directory:

> Available from the publishers, ACM Press or Addison-Wesley Publishing Company.
>
> Public-domain sources for all programs and data of The Stanford GraphBase are freely available. They can be obtained, for example, by anonymous ftp from the master sources on ftp.cs.stanford.edu in directory ~ftp/pub/sgb. The programs are highly portable and have been installed on a wide variety of computers and operating systems.

There is a Unix style "Makefile" to customize the locations of the output files which are generated by the C Compiler. C is available as part of Apple's Xcode integrated development environment.

4) a) Install the TEX files from the following website:
http://tug.org/mactex:
To download, click **MacTeX Download**.
This distribution requires Mac OS 10.13, High Sierra, or higher and runs on Intel processors.
(Note: the mactex-20200407.pkg file is 4.21 GB)

- OR for regular TEX and a much smaller install -

b) To download the smaller BasicTeX, click **Smaller Download**.
(Note: the mactex-basictex-20200407.pkg file is 83.2 MB)
Either one or both versions can be downloaded and reside in different directories.

In the install log and command line text below, <user-name> refers to your Apple user name that appears under the /Users directory. In order to browse the Apple Directory structure you will need to invoke the "Terminal" app in your Application directory by clicking on the Launchpad ICON at the bottom. You can also get to any directory from the Finder app by issuing the keyboard command sequence <shift> <command> <G key> and entering the full path name of the directory you wish to view in the the pop-up menu that is displayed. To see the top level directory, i.e. the root, just enter "/" as the path and then to see the contents of that enter the command "ls" to list the subfiles.

You can also display the environment variables that Mac OS uses such as $PATH, which is list of directories that is searched to find programs or apps scattered across several locations. Launch the Terminal app and enter: SUDO ENV to invoke a restricted command as a "Super User" and hit return followed by your password as follows to display these variables:

```
My-Mac-Air:Knuth <user-name>$ sudo env
Password: <enter your login password>
TERM=xterm-256color
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.MKxIwsbBYG/Listeners
PATH=/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/Library/TeX/texbin
LANG=en_US.UTF-8
HOME=/Users/<user-name>
( . . . and other ENV variables)
```

Note that the $PATH variable has /Library/TeX/texbin as one of the directories (where the TEX app is located). You may also enter the command "cd $HOME/Desktop" or "cd ~/Desktop" to get to your desktop directory. The install script Makefile has terminal commands that compile and install files in difference directories. It has the following phases by entering "make <phase>" in the terminal app:

```
My-Mac-Air:/ <user-name>$ make tests             (checks for existence of install files)
My-Mac-Air:/ <user-name>$ sudo make install      (compiles C source files, and others)
My-Mac-Air:/ <user-name>$ sudo make installdemos (compiles demo programs and installs them)
```

In order to install the apps (binary files) in system directories you must run the command as a Super User (SUDO) and enter the logon password to continue. The Makefile has commands in it that specify the installation directories and you can edit those names to put the files where you wish but it should work OK as is. In this installation the source files are located in /Users/<user-name>/Desktop/Knuth in the subdirectories CWEB and "Stanford GraphBase".

==========

```
My-Mac-Air:desktop <user-name>$ cd /
My-Mac-Air:/ <user-name>$ ls

Applications         etc
Library              home
Network              installer.failurerequests
System               net
Users                private
Volumes              sbin
bin                  tmp
cores                usr
dev                  var
```

==========

Note: If your user-name is Knuth you enter “ls /Users/Knuth”. The files in the Desktop directory appear as Icons on your display.

```
My-Mac-Air:<user-name> <user-name>$ ls /Users/<user-name>

Creative Cloud Files Library     Public
Desktop          Movies          hello
Documents        Music           pslog.txt
Downloads        Pictures        rename
```

==========

Note: Knuth and C Programs are directories. The Knuth directory has the installation files. The other files are not part of the installation. Xcode.app is where Apple’s IDE app resides.

```
My-Mac-Air:desktop <user-name>$ ls /Users/Knuth/Desktop

C Programs                       Knuth
MS_Word_File.docx                My_Software
Xcode.app
```

If you enter: ls -l ~/Desktop for example you will see an extended list of attributes for each file:

```
Tims-Air:desktop timnicholson$ ls -l ~/Desktop
total 62120
drwxr-xr-x     5 timnicholson  staff       160 Aug 27 21:58 C Programs
drwxr-xr-x    21 timnicholson  staff       672 Sep  1 18:39 Knuth
-rw-r--r--@    1 timnicholson  staff   1228647 Sep  1 16:53 MS_Word_File.docx
drwxr-xr-x    15 timnicholson  staff       480 Sep  1 17:49 My_Software
drwxr-xr-x     3 timnicholson  staff        96 Oct 19  2018 Xcode.app
```

==========

```
My-Mac-Air:Knuth <user-name>$ ls /Users/<user-name>/Desktop/Knuth

ERRATA in Stanford Graphbase 2020.rtf
Knuth Vol 4a Errors.pdf
Knuth Vol 4B Facile 8a.ps
Stanford GraphBase
Knuth Vol 4B Facile 9a.ps
TEX Install ReadMe.pdf
Knuth Vol 4a All Errors.pdf
CWEB
```

This directory contains other documents downloaded from Knuth’s webpages including errors in his publications and pre-published versions of Volumes 4B and later 4C. They are not part of the installation.

==========

```
My-Mac-Air:Knuth <user-name>$ cd "Stanford GraphBase"

My-Mac-Air:Stanford GraphBase <user-name>$ ls

+The+Stanford+GraphBase+  gb_basic.w          homer.dat
AMIGA                     gb_books.w          huck.dat
ANSI                      gb_dijk.w           jean.dat
ERRATA                    gb_econ.w           ladders.w
MSVC                      gb_flip.w           lisa.dat
Makefile                  gb_games.w          miles.dat
PROTOTYPES                gb_gates.w          miles_span.w
README                    gb_graph.w          multiply.w
abstract.plaintex         gb_io.w             queen.w
anna.dat                  gb_lisa.w           queen_wrap.ch
assign_lisa.w             gb_miles.w          roget.dat
blank.w                   gb_plane.w          roget_components.w
boilerplate.w             gb_raman.w          sample.correct
book_components.w         gb_rand.w           take_risc.w
cities.texmap             gb_roget.w          test.correct
david.dat                 gb_save.w           test.dat
econ.dat                  gb_sort.w           test_sample.w
econ_order.w              gb_types.w          word_components.w
football.w                gb_words.w          word_giant.ch
```

```
games.dat                girth.w             words.dat

Note: After running the Makefile in this directory additional files will be placed there.
See the section below for the updated list of files.

========================================= Make Tests =========================================

My-Mac-Air:Stanford GraphBase <user-name>$ make tests

if test -r gb_io.ch; then ctangle gb_io.w gb_io.ch; else ctangle gb_io.w; fi
This is CTANGLE (Version 3.64)
*1*8*10*21*28*38*43
Writing the output files: (gb_io.c)..
(test_io.c)
(gb_io.h)
Done.
(No errors were found.)
cc -g -I/usr/local/sgb/include  -DDATA_DIRECTORY=\"/usr/local/sgb/data/\" -c gb_io.c
gb_io.w:201:14: warning: result of comparison of constant 255 with expression of type 'char' is
      always false [-Wtautological-constant-out-of-range-compare]
return(c<0||c> 255)?unexpected_char:icode[c];
            ~^ ~~~
1 warning generated.
cc -g -I/usr/local/sgb/include  test_io.c gb_io.o -o test_io
make gb_graph.c
if test -r gb_graph.ch; then ctangle gb_graph.w gb_graph.ch; else ctangle gb_graph.w; fi
This is CTANGLE (Version 3.64)
*1*8*11*20*42*49
Writing the output files: (gb_graph.c)...
(test_graph.c)
(gb_graph.h).
Done.
(No errors were found.)
make gb_graph.o
cc -g -I/usr/local/sgb/include    -c -o gb_graph.o gb_graph.c
cc -g -I/usr/local/sgb/include  test_graph.c gb_graph.o -o test_graph
make gb_flip.c
if test -r gb_flip.ch; then ctangle gb_flip.w gb_flip.ch; else ctangle gb_flip.w; fi
This is CTANGLE (Version 3.64)
*1*4*8*12*14
Writing the output files: (gb_flip.c)
(test_flip.c)
(gb_flip.h)
Done.
(No errors were found.)
make gb_flip.o
cc -g -I/usr/local/sgb/include    -c -o gb_flip.o gb_flip.c
cc -g -I/usr/local/sgb/include  test_flip.c gb_flip.o -o test_flip
./test_io
OK, the gb_io routines seem to work!
./test_graph

................................................................................
.............................Hey, I allocated 10000000 bytes successfully. Terrific...

OK, the gb_graph routines seem to work!
./test_flip
OK, the gb_flip routines seem to work!
make gb_sort.o
make gb_sort.c
if test -r gb_sort.ch; then ctangle gb_sort.w gb_sort.ch; else ctangle gb_sort.w; fi
This is CTANGLE (Version 3.64)
*1*12
Writing the output files: (gb_sort.c)
(gb_sort.h)
Done.
(No errors were found.)
make gb_sort.o
cc -g -I/usr/local/sgb/include    -c -o gb_sort.o gb_sort.c
make lib
make gb_basic.c
if test -r gb_basic.ch; then ctangle gb_basic.w gb_basic.ch; else ctangle gb_basic.w; fi
This is CTANGLE (Version 3.64)
*1*6*24*36*41*54*63*73*77*87*94*100*115
Writing the output files: (gb_basic.c).....500.....1000.....1500.
(gb_basic.h)
Done.
(No errors were found.)
make gb_basic.o
cc -g -I/usr/local/sgb/include    -c -o gb_basic.o gb_basic.c
make gb_books.c
if test -r gb_books.ch; then ctangle gb_books.w gb_books.ch; else ctangle gb_books.w; fi
This is CTANGLE (Version 3.64)
*1*12*19*26*30
Writing the output files: (gb_books.c)..
(gb_books.h)
Done.
(No errors were found.)
make gb_books.o
cc -g -I/usr/local/sgb/include    -c -o gb_books.o gb_books.c
```

```
make gb_econ.c
if test -r gb_econ.ch; then ctangle gb_econ.w gb_econ.ch; else ctangle gb_econ.w; fi
This is CTANGLE (Version 3.64)
*1*11*17*25*31
Writing the output files: (gb_econ.c)...
(gb_econ.h)
Done.
(No errors were found.)
make gb_econ.o
cc -g -I/usr/local/sgb/include    -c -o gb_econ.o gb_econ.c
make gb_games.c
if test -r gb_games.ch; then ctangle gb_games.w gb_games.ch; else ctangle gb_games.w; fi
This is CTANGLE (Version 3.64)
*1*11*21*25
Writing the output files: (gb_games.c)..
(gb_games.h)
Done.
(No errors were found.)
make gb_games.o
cc -g -I/usr/local/sgb/include    -c -o gb_games.o gb_games.c
make gb_gates.c
if test -r gb_gates.ch; then ctangle gb_gates.w gb_gates.ch; else ctangle gb_gates.w; fi
This is CTANGLE (Version 3.64)
*1*8*38*43*49*66*75*84*86
Writing the output files: (gb_gates.c).....500.....1000..
(gb_gates.h)
Done.
(No errors were found.)
make gb_gates.o
cc -g -I/usr/local/sgb/include    -c -o gb_gates.o gb_gates.c
gb_gates.w:1112:8: warning: format string is not a string literal (potentially insecure)
      [-Wformat-security]
printf(a->tip->name);
       ^~~~~~~~~~~~
gb_gates.w:1112:8: note: treat the string as an argument to avoid this
printf(a->tip->name);
       ^
       "%s",
1 warning generated.
make gb_lisa.c
if test -r gb_lisa.ch; then ctangle gb_lisa.w gb_lisa.ch; else ctangle gb_lisa.w; fi
This is CTANGLE (Version 3.64)
*1*11*15*19*23*33*37
Writing the output files: (gb_lisa.c)...
(gb_lisa.h)
Done.
(No errors were found.)
make gb_lisa.o
cc -g -I/usr/local/sgb/include    -c -o gb_lisa.o gb_lisa.c
make gb_miles.c
if test -r gb_miles.ch; then ctangle gb_miles.w gb_miles.ch; else ctangle gb_miles.w; fi
This is CTANGLE (Version 3.64)
*1*9*17*22
Writing the output files: (gb_miles.c)..
(gb_miles.h)
Done.
(No errors were found.)
make gb_miles.o
cc -g -I/usr/local/sgb/include    -c -o gb_miles.o gb_miles.c
make gb_plane.c
if test -r gb_plane.ch; then ctangle gb_plane.w gb_plane.ch; else ctangle gb_plane.w; fi
This is CTANGLE (Version 3.64)
*1*8*13*20*25*34*41*45
Writing the output files: (gb_plane.c).....500
(gb_plane.h)
Done.
(No errors were found.)
make gb_plane.o
cc -g -I/usr/local/sgb/include    -c -o gb_plane.o gb_plane.c
make gb_raman.c
if test -r gb_raman.ch; then ctangle gb_raman.w gb_raman.ch; else ctangle gb_raman.w; fi
This is CTANGLE (Version 3.64)
*1*6*13*18*26*30*32
Writing the output files: (gb_raman.c)...
(gb_raman.h)
Done.
(No errors were found.)
make gb_raman.o
cc -g -I/usr/local/sgb/include    -c -o gb_raman.o gb_raman.c
make gb_rand.c
if test -r gb_rand.ch; then ctangle gb_rand.w gb_rand.ch; else ctangle gb_rand.w; fi
This is CTANGLE (Version 3.64)
*1*11*22*24*28
Writing the output files: (gb_rand.c)...
(gb_rand.h)
Done.
(No errors were found.)
make gb_rand.o
cc -g -I/usr/local/sgb/include    -c -o gb_rand.o gb_rand.c
gb_rand.w:228:1: warning: add explicit braces to avoid dangling else [-Wdangling-else]
```

```
else{
^
gb_rand.w:223:1: warning: add explicit braces to avoid dangling else [-Wdangling-else]
else{register Arc*a;
^
2 warnings generated.
make gb_roget.c
if test -r gb_roget.ch; then ctangle gb_roget.w gb_roget.ch; else ctangle gb_roget.w; fi
This is CTANGLE (Version 3.64)
*1*6*10*15
Writing the output files: (gb_roget.c).
(gb_roget.h)
Done.
(No errors were found.)
make gb_roget.o
cc -g -I/usr/local/sgb/include    -c -o gb_roget.o gb_roget.c
make gb_words.c
if test -r gb_words.ch; then ctangle gb_words.w gb_words.ch; else ctangle gb_words.w; fi
This is CTANGLE (Version 3.64)
*1*9*14*22*30*32
Writing the output files: (gb_words.c)...
(gb_words.h)
Done.
(No errors were found.)
make gb_words.o
cc -g -I/usr/local/sgb/include    -c -o gb_words.o gb_words.c
make gb_dijk.c
if test -r gb_dijk.ch; then ctangle gb_dijk.w gb_dijk.ch; else ctangle gb_dijk.w; fi
This is CTANGLE (Version 3.64)
*1*4*15*20*26
Writing the output files: (gb_dijk.c)..
(gb_dijk.h)
Done.
(No errors were found.)
make gb_dijk.o
cc -g -I/usr/local/sgb/include    -c -o gb_dijk.o gb_dijk.c
make gb_save.c
if test -r gb_save.ch; then ctangle gb_save.w gb_save.ch; else ctangle gb_save.w; fi
This is CTANGLE (Version 3.64)
*1*3*19*47
Writing the output files: (gb_save.c).....500.
(gb_save.h)
Done.
(No errors were found.)
make gb_save.o
cc -g -I/usr/local/sgb/include    -c -o gb_save.o gb_save.c
rm -f certified
ar rcv libgb.a gb_flip.o gb_graph.o gb_io.o gb_sort.o gb_basic.o gb_books.o gb_econ.o
gb_games.o gb_gates.o gb_lisa.o gb_miles.o gb_plane.o gb_raman.o gb_rand.o gb_roget.o
gb_words.o gb_dijk.o gb_save.o
a - gb_flip.o
a - gb_graph.o
a - gb_io.o
a - gb_sort.o
a - gb_basic.o
a - gb_books.o
a - gb_econ.o
a - gb_games.o
a - gb_gates.o
a - gb_lisa.o
a - gb_miles.o
a - gb_plane.o
a - gb_raman.o
a - gb_rand.o
a - gb_roget.o
a - gb_words.o
a - gb_dijk.o
a - gb_save.o
ranlib libgb.a
make test_sample
make test_sample.c
if test -r test_sample.ch; then ctangle test_sample.w test_sample.ch; else ctangle test_sample.w; fi
This is CTANGLE (Version 3.64)
*1*13*19
Writing the output file (test_sample.c):.
Done.
(No errors were found.)
make test_sample
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  test_sample.c -lgb -lgb -o test_sample
test_sample.w:113:5: warning: using the result of an assignment as a condition without
      parentheses [-Wparentheses]
if(i= random_lengths(g,0L,10L,12L,dst,2L))
   ~^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
test_sample.w:113:5: note: place parentheses around the assignment to silence this warning
if(i= random_lengths(g,0L,10L,12L,dst,2L))
    ^
   (                                  )
test_sample.w:113:5: note: use '==' to turn this assignment into an equality comparison
if(i= random_lengths(g,0L,10L,12L,dst,2L))
    ^
```

```
    ==
1 warning generated.
./test_sample > sample.out
diff test.gb test.correct
diff sample.out sample.correct
rm test.gb sample.out test_io test_graph test_flip test_sample
echo "Congratulations --- the tests have all been passed."
Congratulations --- the tests have all been passed.
touch certified

================================================ Make Install ==================================

My-Mac-Air:Stanford GraphBase <user-name>$ make install
if test ! -r certified; then echo "Please run 'make tests' first!"; fi
test -r certified
make installdata
mkdir /usr/local/sgb
mkdir: /usr/local/sgb: File exists
make[1]: [installdata] Error 1 (ignored)
mkdir /usr/local/sgb/data
mkdir: /usr/local/sgb/data: Permission denied
make[1]: [installdata] Error 1 (ignored)
cp -p anna.dat david.dat econ.dat games.dat homer.dat huck.dat jean.dat lisa.dat miles.dat roget.dat words.dat /usr/local/sgb/
data
usage: cp [-R [-H | -L | -P]] [-fi | -n] [-apvXc] source_file target_file
       cp [-R [-H | -L | -P]] [-fi | -n] [-apvXc] source_file ... target_directory
make[1]: [installdata] Error 64 (ignored)
mkdir /usr/local/lib
mkdir: /usr/local/lib: File exists
make: [install] Error 1 (ignored)
cp libgb.a /usr/local/lib
mkdir /usr/local/lib/cweb
mkdir: /usr/local/lib/cweb: File exists
make: [install] Error 1 (ignored)
cp -p boilerplate.w gb_types.w /usr/local/lib/cweb
mkdir /usr/local/sgb/include
mkdir: /usr/local/sgb/include: Permission denied
make: [install] Error 1 (ignored)
cp -p gb_flip.h gb_graph.h gb_io.h gb_sort.h gb_basic.h gb_books.h gb_econ.h gb_games.h gb_gates.h gb_lisa.h gb_miles.h
gb_plane.h gb_raman.h gb_rand.h gb_roget.h gb_words.h gb_dijk.h gb_save.h Makefile /usr/local/sgb/include
usage: cp [-R [-H | -L | -P]] [-fi | -n] [-apvXc] source_file target_file
       cp [-R [-H | -L | -P]] [-fi | -n] [-apvXc] source_file ... target_directory
make: [install] Error 64 (ignored)

====================== Make Install Failed - Rerun With Super User “SUDO” Cmd ===================


My-Mac-Air:Stanford GraphBase <user-name>$ sudo make install
Password: <enter your login password>

if test ! -r certified; then echo "Please run 'make tests' first!"; fi
test -r certified
make installdata
mkdir /usr/local/sgb
mkdir: /usr/local/sgb: File exists
make[1]: [installdata] Error 1 (ignored)
mkdir /usr/local/sgb/data
cp -p anna.dat david.dat econ.dat games.dat homer.dat huck.dat jean.dat lisa.dat miles.dat roget.dat words.dat /usr/local/sgb/
data
mkdir /usr/local/lib
mkdir: /usr/local/lib: File exists
make: [install] Error 1 (ignored)
cp libgb.a /usr/local/lib
mkdir /usr/local/lib/cweb
mkdir: /usr/local/lib/cweb: File exists
make: [install] Error 1 (ignored)
cp -p boilerplate.w gb_types.w /usr/local/lib/cweb
mkdir /usr/local/sgb/include
cp -p gb_flip.h gb_graph.h gb_io.h gb_sort.h gb_basic.h gb_books.h gb_econ.h gb_games.h
gb_gates.h gb_lisa.h gb_miles.h gb_plane.h gb_raman.h gb_rand.h gb_roget.h gb_words.h
gb_dijk.h gb_save.h Makefile /usr/local/sgb/include

================================================================================================

Note: The source directory has been updated by the “Make Install”
commands that added the <program>.c and <program>.o and other files and
the symbolic links (i.e. program-name.dSYM) to other directories.

cd /Users/<user-name>/Desktop/Knuth/Stanford GraphBase

My-Mac-Air:Stanford GraphBase <user-name>$ ls

+The+Stanford+GraphBase+  gb_plane.h
AMIGA                     gb_plane.o
ANSI                      gb_plane.w
ERRATA                    gb_raman.c
MSVC                      gb_raman.h
Makefile                  gb_raman.o
PROTOTYPES                gb_raman.w
README                    gb_rand.c
```

```
abstract.plaintex        gb_rand.h
anna.dat                 gb_rand.o
assign_lisa.c            gb_rand.w
assign_lisa.dSYM         gb_roget.c
assign_lisa.w            gb_roget.h
blank.w                  gb_roget.o
boilerplate.w            gb_roget.w
book_components.c        gb_save.c
book_components.dSYM     gb_save.h
book_components.w        gb_save.o
certified                gb_save.w
cities.texmap            gb_sort.c
david.dat                gb_sort.h
econ.dat                 gb_sort.o
econ_order.c             gb_sort.w
econ_order.dSYM          gb_types.w
econ_order.w             gb_words.c
football.c               gb_words.h
football.dSYM            gb_words.o
football.w               gb_words.w
games.dat                girth.c
gb_basic.c               girth.dSYM
gb_basic.h               girth.w
gb_basic.o               homer.dat
gb_basic.w               huck.dat
gb_books.c               jean.dat
gb_books.h               ladders.c
gb_books.o               ladders.dSYM
gb_books.w               ladders.w
gb_dijk.c                libgb.a
gb_dijk.h                lisa.dat
gb_dijk.o                miles.dat
gb_dijk.w                miles_span.c
gb_econ.c                miles_span.dSYM
gb_econ.h                miles_span.w
gb_econ.o                multiply.c
gb_econ.w                multiply.dSYM
gb_flip.c                multiply.w
gb_flip.h                queen.c
gb_flip.o                queen.dSYM
gb_flip.w                queen.w
gb_games.c               queen_wrap.ch
gb_games.h               roget.dat
gb_games.o               roget_components.c
gb_games.w               roget_components.dSYM
gb_gates.c               roget_components.w
gb_gates.h               sample.correct
gb_gates.o               take_risc.c
gb_gates.w               take_risc.dSYM
gb_graph.c               take_risc.w
gb_graph.h               test.correct
gb_graph.o               test.dat
gb_graph.w               test_flip.c
gb_io.c                  test_flip.dSYM
gb_io.h                  test_graph.c
gb_io.o                  test_graph.dSYM
gb_io.w                  test_io.c
gb_lisa.c                test_io.dSYM
gb_lisa.h                test_sample.c
gb_lisa.o                test_sample.dSYM
gb_lisa.w                test_sample.w
gb_miles.c               word_components.c
gb_miles.h               word_components.dSYM
gb_miles.o               word_components.w
gb_miles.w               word_giant.ch
gb_plane.c               words.dat

====================================== Make Installdemos ======================================

My-Mac-Air:Stanford GraphBase <user-name>$ pwd
/users/<user-name>/desktop/Knuth/Stanford GraphBase

My-Mac-Air:Stanford GraphBase <user-name>$ sudo make installdemos
Password:
make assign_lisa.c
if test -r assign_lisa.ch; then ctangle assign_lisa.w assign_lisa.ch; else ctangle assign_lisa.w; fi
This is CTANGLE (Version 3.64)
*1*8*14*24*28*32
Writing the output file (assign_lisa.c):...
Done.
(No errors were found.)
make assign_lisa
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  assign_lisa.c -lgb -lgb -o assign_lisa
assign_lisa.w:73:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
1 warning generated.
make book_components.c
if test -r book_components.ch; then ctangle book_components.w book_components.ch; else ctangle book_components.w; fi
This is CTANGLE (Version 3.64)
```

```
*1*6*21*24
Writing the output file (book_components.c):.
Done.
(No errors were found.)
make book_components
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  book_components.c -lgb -lgb -o book_components
book_components.w:58:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
1 warning generated.
make econ_order.c
if test -r econ_order.ch; then ctangle econ_order.w econ_order.ch; else ctangle econ_order.w; fi
This is CTANGLE (Version 3.64)
*1*7*15
Writing the output file (econ_order.c):.
Done.
(No errors were found.)
make econ_order
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  econ_order.c -lgb -lgb -o econ_order
econ_order.w:80:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
1 warning generated.
make football.c
if test -r football.ch; then ctangle football.w football.ch; else ctangle football.w; fi
This is CTANGLE (Version 3.64)
*1*6*8*19*26*36
Writing the output file (football.c):...
Done.
(No errors were found.)
make football
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  football.c -lgb -lgb -o football
football.w:61:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
football.w:296:1: warning: add explicit braces to avoid dangling else [-Wdangling-else]
else best_arc= a,d= a->del;
^
2 warnings generated.
make girth.c
if test -r girth.ch; then ctangle girth.w girth.ch; else ctangle girth.w; fi
This is CTANGLE (Version 3.64)
*1*6*12*14
Writing the output file (girth.c):.
Done.
(No errors were found.)
make girth
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  girth.c -lgb -lgb -o girth
girth.w:65:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main()
^
1 warning generated.
make ladders.c
if test -r ladders.ch; then ctangle ladders.w ladders.ch; else ctangle ladders.w; fi
This is CTANGLE (Version 3.64)
*1*4*6*12*26*28
Writing the output file (ladders.c):..
Done.
(No errors were found.)
make ladders
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  ladders.c -lgb -lgb -o ladders
ladders.w:91:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
ladders.w:253:18: warning: format specifies type 'long' but the argument has type 'int'
      [-Wformat]
quit_if(gg==NULL,no_room+5);
~~~~~~~~~~~~~~~~~^~~~~~~~~
gb_graph.w:107:17: note: expanded from macro 'no_room'
#define no_room 1
                ^
ladders.w:91:62: note: expanded from macro 'quit_if'
"Sorry, I couldn't build a dictionary (trouble code %ld)!\n",c) ; \
                                                        ~~~      ^
ladders.w:259:25: warning: format specifies type 'long' but the argument has type 'int'
      [-Wformat]
quit_if(gb_trouble_code,no_room+6);
~~~~~~~~~~~~~~~~~~~~~~~~^~~~~~~~~
gb_graph.w:107:17: note: expanded from macro 'no_room'
#define no_room 1
                ^
ladders.w:91:62: note: expanded from macro 'quit_if'
"Sorry, I couldn't build a dictionary (trouble code %ld)!\n",c) ; \
                                                        ~~~      ^
3 warnings generated.
make miles_span.c
if test -r miles_span.ch; then ctangle miles_span.w miles_span.ch; else ctangle miles_span.w; fi
This is CTANGLE (Version 3.64)
*1*8*12*19*23*29*43*55*64*71*72
Writing the output file (miles_span.c):.....500...
```

```
Done.
(No errors were found.)
make miles_span
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  miles_span.c -lgb -lgb -o miles_span
miles_span.w:1155:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
qunite(m,q,mm,qq,h)
^
miles_span.w:1176:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
miles_span.w:1257:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
qenque(h,a)
^
miles_span.w:1265:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
miles_span.w:1272:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
qmerge(h,hh)
^
miles_span.w:1284:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
miles_span.w:1339:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
qtraverse(h,visit)
^
miles_span.w:1362:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
miles_span.w:197:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
report(u,v,l)
^
miles_span.w:202:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
miles_span.w:99:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
11 warnings generated.
make multiply.c
if test -r multiply.ch; then ctangle multiply.w multiply.ch; else ctangle multiply.w; fi
This is CTANGLE (Version 3.64)
*1*10*13*16
Writing the output file (multiply.c):..
Done.
(No errors were found.)
make multiply
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  multiply.c -lgb -lgb -o multiply
multiply.w:200:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
decimal_to_binary(x,s,n)
^
multiply.w:222:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
multiply.w:38:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
multiply.w:183:69: warning: format specifies type 'int' but the argument has type 'long'
      [-Wformat]
printf("Please try another seed value; %d makes the answer zero!\n",seed);
                                       ~~                           ^~~~
                                       %ld
4 warnings generated.
make queen.c
if test -r queen.ch; then ctangle queen.w queen.ch; else ctangle queen.w; fi
This is CTANGLE (Version 3.64)
*1*3
Writing the output file (queen.c):
Done.
(No errors were found.)
make queen
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  queen.c -lgb -lgb -o queen
queen.w:26:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main()
^
1 warning generated.
make roget_components.c
if test -r roget_components.ch; then ctangle roget_components.w roget_components.ch; else ctangle roget_components.w; fi
This is CTANGLE (Version 3.64)
*1*18
Writing the output file (roget_components.c):.
Done.
(No errors were found.)
make roget_components
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  roget_components.c -lgb -lgb -o roget_components
roget_components.w:45:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
1 warning generated.
make take_risc.c
if test -r take_risc.ch; then ctangle take_risc.w take_risc.ch; else ctangle take_risc.w; fi
```

```
This is CTANGLE (Version 3.64)
*1*6*9
Writing the output file (take_risc.c):.
Done.
(No errors were found.)
make take_risc
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  take_risc.c -lgb -lgb -o take_risc
take_risc.w:35:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main(argc,argv)
^
1 warning generated.
make word_components.c
if test -r word_components.ch; then ctangle word_components.w word_components.ch; else ctangle word_components.w; fi
This is CTANGLE (Version 3.64)
*1*6
Writing the output file (word_components.c):
Done.
(No errors were found.)
make word_components
cc -g -I/usr/local/sgb/include   -L. -L/usr/local/lib  word_components.c -lgb -lgb -o word_components
word_components.w:19:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
main()
^
1 warning generated.
mkdir /usr/local/bin
mkdir: /usr/local/bin: File exists
make: [installdemos] Error 1 (ignored)
mv assign_lisa book_components econ_order football girth ladders miles_span multiply queen
   roget_components take_risc word_components /usr/local/bin

================================ End of Stanford GraphBase Install Log ==========================

======================================== Updated Directories ====================================

My-Mac-Air:desktop <user-name>$ cd /
My-Mac-Air:/ <user-name>$ ls

Applications         etc
Library              home
Network              installer.failurerequests
System               net
Users                private
Volumes              sbin
bin                  tmp
cores                usr
dev                  var

=========================================================================================================

My-Mac-Air:usr <user-name>$ ls /usr

bin          lib          local        share
include      libexec      sbin         standalone

=========================================================================================================

Note: sgb in blue text indicates this is the GraphBase directory

My-Mac-Air:local <user-name>$ ls /usr/local

Caskroom             Homebrew         include      remotedesktop
share                Cellar           bin          lib
sbin                 texlive          Frameworks   etc
opt                  sgb              var

=========================================================================================================

My-Mac-Air:sgb <user-name>$ ls /usr/local/sgb

data     include

=========================================================================================================

This directory has the data files for the Stanford GraphBase programs.

My-Mac-Air:sgb <user-name>$ ls /usr/local/sgb/data

anna.dat     games.dat    jean.dat     roget.dat
david.dat    homer.dat    lisa.dat     words.dat
econ.dat     huck.dat     miles.dat

=========================================================================================================

This directory has the “include <filename.h>” files for the Stanford GraphBase programs
Which are incorporated into the C program files when they are compiled.

My-Mac-Air:sgb <user-name>$ ls /usr/local/sgb/include

Makefile     gb_econ.h    gb_graph.h   gb_plane.h   gb_save.h
gb_basic.h   gb_flip.h    gb_io.h      gb_raman.h   gb_sort.h
```

```
gb_books.h   gb_games.h   gb_lisa.h    gb_rand.h    gb_words.h
gb_dijk.h    gb_gates.h   gb_miles.h   gb_roget.h

================================================================================================

My-Mac-Air:local <user-name>$ ls /usr/local/bin

Note: Stanford GraphBase Programs - Are in blue text, as well as the
CWEB binaries ctangle and cweave.

assign_lisa          gsdj500          printafm
book_components      gslj             ps2ascii
brew                 gslp             ps2epsi
cordova              gsnd             ps2pdf
ctangle              ios-deploy       ps2pdf12
cweave               ladders          ps2pdf13
dvipdf               lprsetup.sh      ps2pdf14
econ_order           miles_span       ps2pdfwr
eps2eps              multiply         ps2ps
football             node             ps2ps2
fuzzy_match          npm              queen
girth                pdf2dsc          roget_components
gs                   pdf2ps           sandbox-pod
gs-X11               pf2afm           take_risc
gs-noX11             pfbtopfa         unix-lpr.sh
gsbj                 pod              word_components
gsdj                 pphs             xcodeproj

====================== Stanford GraphBase - Test Results ====================

Chapter 1.1 WORDS (page 2 of 1994 version)
See:   The Stanford GraphBase - A Platform for Combinatorial Computing
       By Donald E. Knuth, Copyright 1994 by the ACM Press

My-Mac-Air:Stanford GraphBase <user-name>$ ladders

Starting word: words
    Goal word: graph
         0 words
         1 wolds
         2 golds
         3 goads
         4 grads
         5 grade
         6 grape
         7 graph

Starting word: order
    Goal word: slobs
         0 order
         1 odder
         2 adder
         3 aider
         4 aides
         5 sides
         6 sires
         7 sores
         8 sorts
         9 soots
        10 slots
        11 slobs

Starting word: chaos
    Goal word: slobs
         0 chaos
         1 choos
         2 chows
         3 shows
         4 slows
         5 slobs

================================================================================================

Chapter 1.3 BOOKS (page 12 of 1994 version)

My-Mac-Air:Stanford GraphBase <user-name>$ book_components -tanna -n10 -f15 -l20 -V

Biconnectivity analysis of book("anna",10,0,15,20,1,1,0)

LE=Konstantin Dmitrievitch Levin, proprietor of Pokrovskoe [weight 103]
AN=Anna Arkadyevna Karenina, wife of AL [weight 73]
VR=Count Alexey Kirillovitch Vronsky, young officer [weight 67]
ST=Prince Stepan Arkadyevitch Oblonsky (Stiva), brother of AN [weight 64]
KI=Princess Ekaterina Alexandrovna Shtcherbatskaya (Kitty), wife of LE [weight 59]
DO=Princess Darya Alexandrovna Oblonskaya (Dolly), wife of ST [weight 46]
AL=Alexey Alexandrovitch Karenin, minister of state [weight 39]
KO=Sergei Ivanovitch Koznishev, half-brother of LE [weight 38]
PS=Princess Shtcherbatskaya, mother of DO and KI [weight 27]
```

```
PR=Prince Alexander Shtcherbatsky, father of DO and KI [weight 25]

Isolated vertex LE
Bicomponent PR also includes:
 PS (from PR; ..to KI)
 and articulation point KI
Bicomponent KI also includes:
 VR (from ST; ..to AN)
 DO (from ST; ..to AN)
 ST (from KI; ..to AN)
 and AN (this ends a connected component of the graph)
Isolated vertex AL
Isolated vertex KO
```

Chapter 1.10 GATES (page 32 of 1994 version)

Note: The “multiply” command does not accept 100 bit numbers or even 40 bit numbers.

```
My-Mac-Air:Knuth <user-name>$ multiply 20 20

Here I am, ready to multiply 20-bit numbers by 20-bit numbers.
(I'm simulating a logic circuit with 2820 gates, depth 27.)

Number, please? <hit return to exit>

My-Mac-Air:Knuth <user-name>$ multiply 40 40
Abort trap: 6

My-Mac-Air:Knuth <user-name>$ multiply 30 30
Here I am, ready to multiply 30-bit numbers by 30-bit numbers.
(I'm simulating a logic circuit with 6177 gates, depth 32.)

Number, please? 123456789
Another? 100000001
123456789x100000001=12345679023456789.
```