Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Hello, world!

```latex
\documentclass{article}

% All LaTeX documents including
% tikz() output must use this
% package!
\usepackage{tikz}

\begin{document}
  \begin{figure}[!h]
    \centering

    % The output from tikz()
    % is imported here.
    \input{simpleEx.tex}

    \caption{Simple Example}
  \end{figure}
\end{document}
```

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Example of input from a file

```c
/*
 * Copyright (c) 2000 Dan Papasian.  All rights reserved.
 * Ported to MinGW/MSYS in 2011 by Charlie Sharpsteen
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 *
 * 3. The name of the author may not be used to endorse or promote
 * products derived from this software without specific prior written
 * permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR
 * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY
 * DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE
 * GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER
 * IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
 * OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
 * IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#include <sys/stat.h>
#include <sys/param.h>

#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>


static void     usage(void);
static int      print_matches(char *, char *);
static void     to_msys_path(char *path);
static char    *strsep(char **stringp, const char *delim);

int             silent = 0, allpaths = 0, msys = 0;
#define WIN_EXE_SUFFIXES ":.exe:.bat:.com"

int
main(int argc, char **argv)
{
```

```
char            *p, *path;
ssize_t          pathlen;
int              opt, status;

status = EXIT_SUCCESS;

//Test for existance
  of $_.If it is non - NULL, we are probably not being
    // executed by cmd.exe and so will set MSYS to true.
    if (getenv("_") != NULL)
    msys = 1;

while ((opt = getopt(argc, argv, "asmw")) != -1) {
  switch (opt) {
  case 'a':
    allpaths = 1;
    break;
  case 'm':
    msys = 1;
    break;
  case 's':
    silent = 1;
    break;
  case 'w':
    msys = 0;
    break;
  default:
    usage();
    break;
  }
}

argv += optind;
argc -= optind;

if (argc == 0)
  usage();

if ((p = getenv("PATH")) == NULL)
  exit(EXIT_FAILURE);
pathlen = strlen(p) + 1;
path = malloc(pathlen);
if (path == NULL)
  exit(EXIT_FAILURE);

  while (argc > 0) {
    memcpy(path, p, pathlen);

    if (
  strlen(*argv) >= FILENAME_MAX ||
  print_matches(path, *argv) == -1)
      status = EXIT_FAILURE;

    argv++;
    argc--;
  }

  exit(status);
}

static void
usage(void)
{

  (void) fprintf(stderr, "usage: which [-amsw] program ...\n");
  exit(EXIT_FAILURE);
}

static int
is_there(char *filename)
{
  struct stat     fin;
  const char     *suffix;
  char           *suffix_list = NULL;
  char            candidate[PATH_MAX];

  if (suffix_list == NULL)
    suffix_list = (char *) malloc(sizeof(WIN_EXE_SUFFIXES) + 1);
  strcpy(suffix_list, WIN_EXE_SUFFIXES);

  while ((suffix = strsep(&suffix_list, ":")) != NULL) {
    if ((int) (sizeof(filename) + sizeof(suffix)) >= PATH_MAX)
      continue;
    snprintf(candidate, sizeof(candidate), "%s%s", filename, suffix);

    if (
  access(candidate, X_OK) == 0 &&
  stat(candidate, &fin) == 0 &&
  S_ISREG(fin.st_mode)
      ) {
```

```
	if (!silent) {
  if (msys)
    to_msys_path(candidate);
  printf("%s\n", candidate);
	}
	return (1);
    }
  }

  return (0);
}

static int
print_matches(char *path, char *filename)
{
  char		candidate[PATH_MAX];
  const char	*d;
  int		found;

  if (strchr(filename, '/') != NULL)
    return (is_there(filename) ? 0 : -1);
  found = 0;
  while ((d = strsep(&path, ";")) != NULL) {
    if (*d == '\0')
      d = ".";
    if (snprintf(candidate, sizeof(candidate), "%s\\%s", d,
      filename) >= (int) sizeof(candidate))
	continue;
    if (is_there(candidate)) {
      found = 1;
      if (!allpaths)
  break;
    }
  }
  return (found ? 0 : -1);
}

static void
to_msys_path(char *path)
{
  /*
   * Take a Windows-style path such as `C:\foo\bar` and transform it
   * into something MSYS expects: `/c/foo/bar`
   */

  /* Convert Windows drive names to MSYS drive names */
  if ((strlen(path) > 1) && isalpha((int) path[0]) && (path[1] == ':')) {
    path[1] = tolower((int) path[0]);
    path[0] = '/';
  }
  /* Replace Windows path separators with UNIX separators */
  char           *c;
  for (c = path; *c != '\0'; ++c)
    if (*c == '\\')
      *c = '/';
}


/*
 * The following code is taken from strsep.c in the FreeBSD source. It
 * is included here, flagged as static, to guard against linker errors
 * if MinGW happens to add `strsep` to the standard library in a future
 * version.
 */

/*
 * Copyright (c) 1990, 1993 The Regents of the University of California.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met: 1. Redistributions of source code must retain the above
 * copyright notice, this list of conditions and the following
 * disclaimer.  2. Redistributions in binary form must reproduce the
 * above copyright notice, this list of conditions and the following
 * disclaimer in the documentation and/or other materials provided with
 * the distribution.  4. Neither the name of the University nor the
 * names of its contributors may be used to endorse or promote products
 * derived from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS''
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
 * PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
```

```c
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

/*
 * Get next token from string *stringp, where tokens are possibly-empty
 * strings separated by characters from delim.
 *
 * Writes NULs into the string at *stringp to end tokens.  delim need
 * not remain constant from call to call.  On return, *stringp points
 * past the last NUL written (if there might be further tokens), or is
 * NULL (if there are definitely no more tokens).
 *
 * If *stringp is NULL, strsep returns NULL.
 */
static char    *
strsep(stringp, delim)
  char           **stringp;
  const char      *delim;
{
  char            *s;
  const char      *spanp;
  int              c, sc;
  char            *tok;

  if ((s = *stringp) == NULL)
    return (NULL);
  for (tok = s;;) {
    c = *s++;
    spanp = delim;
    do {
      if ((sc = *spanp++) == c) {
  if (c == 0)
    s = NULL;
  else
    s[-1] = 0;
  *stringp = s;
  return (tok);
      }
    } while (sc != 0);
  }
  /* NOTREACHED */
}
```

All done!