```
/* ref. [USP] Chapter 2, Program 2.2, p. 37 */

#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define DELIMITERS "\t"
int makeargv(const char* s, const char* delimiters, char*** argvp) {
   int error;
   int i;
   int numtokens;
   const char* snew;
   char* t;

   if ((s == NULL) || (delimiters == NULL) || (argvp == NULL)) {
      errno = EINVAL;
      return -1;
   }
   *argvp = NULL;
   snew = s + strspn(s, delimiters);         /* snew is real start of string */
   if ((t = malloc(strlen(snew) + 1)) == NULL)
      return -1;
   strcpy(t, snew);
   numtokens = 0;
   if (strtok(t, delimiters) != NULL)     /* count the number of tokens in s */
      for (numtokens = 1; strtok(NULL, delimiters) != NULL; numtokens++) ;

                              /* create argument array for ptrs to the tokens */
   if ((*argvp = malloc((numtokens + 1)*sizeof(char*))) == NULL) {
      error = errno;
      free(t);
      errno = error;
      return -1;
   }
                         /* insert pointers to tokens into the argument array */
   if (numtokens == 0)
      free(t);
   else {
      strcpy(t, snew);
      **argvp = strtok(t, delimiters);
      for (i = 1; i < numtokens; i++)
          *((*argvp) + i) = strtok(NULL, delimiters);
    }
    *((*argvp) + numtokens) = NULL;             /* put in final NULL pointer */
    return numtokens;
}

/* ref. [USP] Chapter 2, Example 2.19, p. 38 */
void freeargv(char** argv) {
   if (argv == NULL)
      return;
   if (*argv != NULL)
      free(*argv);
   free(argv);
}

/* ref. [USP] Chapter 2, Program 2.1, p. 34 */

/* make sure you quote the single command-line arugment to this program, e.g.,
   $ ./a.out "Parse this string into an argument vector."
 */
int argvmain(int argc, char** argv) {

   int i, numtokens;
   char** myargv;

//   if (argc != 2) {
//      fprintf(stderr, "Usage: %s string\n", argv[0]);
//      return 1;
```



```
//   }

   if ((numtokens = makeargv(argv[1], DELIMITERS, &myargv)) == -1) {
      fprintf(stderr, "Failed to construct an argument array for %s.\n", argv[1]);
      return 1;
   }
   printf ("The constructed argument array contains:\n");
   for (i=0; i < numtokens; i++)
      printf("%d:%s:\n", i, myargv[i]);

   execvp(myargv[0], &myargv[0]);
   freeargv(myargv);

   return 0;
}
```