Lab #6, CS 262

Strings and File I/O

GEORGE MASON UNIVERSITY

Where Innovation Is Tradition

Things to Do

- Prompt user for the input and output file
- Open input and output file
  - use file I/O
- Read one line and ask user to delete a word
  - Use strtok()
- Write a new sentence to output file
  - Use strcat()
- Write deleted words to an array
- Close input and output files
- Print out the deleted words

Reading from a text file

```c
#include <stdio.h>
int main()
{
    int num;
    FILE *fptr;

    if ((fptr = fopen("C:\\program.txt","r")) == NULL){
        printf("Error! opening file");

        // Program exits if the file pointer returns NULL.
        exit(1);
    }

    fscanf(fptr,"%d", &num);

    printf("Value of n=%d", num);
    fclose(fptr);

    return 0;
}
```

Writing to a text file

```c
#include <stdio.h>
int main()
{
    int num;
    FILE *fptr;
    fptr = fopen("C:\\program.txt","w");

    if(fptr == NULL)
    {
        printf("Error!");
        exit(1);
    }

    printf("Enter num: ");
    scanf("%d",&num);

    fprintf(fptr,"%d",num);
    fclose(fptr);

    return 0;
}
```

getline

ssize_t getline(char **_lineptr_, size_t *_n_, FILE *_stream_);

```
int
main(void)
{
    FILE *fp;
    char *line = NULL;
    size_t len = 0;
    ssize_t read;

   fp = fopen("/etc/motd", "r");
    if (fp == NULL)
        exit(EXIT_FAILURE);

   while ((read = getline(&line, &len, fp)) != -1) {
        printf("Retrieved line of length %zu :\n", read);
        printf("%s", line);
    }

   free(line);
    exit(EXIT_SUCCESS);
}
```

strtok

```
char *strtok(char *str, const char *delim);
```

str − The contents of this string are modified and broken into smaller strings (tokens)
delim − This is the C string containing the delimiters. These may vary from one call to another

strtok example

```c
#include <string.h>
#include <stdio.h>

int main () {
   char str[80] = "This is CS262 Lab6.";
   const char s[2] = "-";
   char *token;

   /* get the first token */
   token = strtok(str, s);

   /* walk through other tokens */
   while( token != NULL ) {
      printf( " %s\n", token );

      token = strtok(NULL, s);
   }

   return(0);
}
```

strcat

char *strcat(char *dest, const char *src);

dest − This is pointer to the destination array, which should contain a C string, and should be large enough to contain the concatenated resulting string.
src − This is the string to be appended. This should not overlap the destination.

strcat example

```c
#include <stdio.h>
#include <string.h>

int main () {
   char src[50], dest[50];

   strcpy(src,  "This is source");
   strcpy(dest, "This is destination");

   strcat(dest, src);

   printf("Final destination string : |%s|", dest);

   return(0);
}
```