ForNextDay(9)
Stephen Cole
3553803

9a

printargs.c

```
#include<stdio.h>

int main(int argc,char* argv[])
{
   int i;
   printf("Number Of Arguments Passed: %d\n",argc);
         for(i=0;i<argc;i++)
         printf("argv[%d]: %p\n", i, &argv[i]);
   return 0;
}
```

```
~/Documents/courses/cs2263/lecture/lecture9 $ ./args One Two Three

Number Of Arguments Passed: 4

argv[0]: 0x7ffeef945880

argv[1]: 0x7ffeef945888

argv[2]: 0x7ffeef945890

argv[3]: 0x7ffeef945898
```

9b

Using the project's Makefile, explain the purpose of:

- $@ symbol
- $(CFLAGS)

$@ is used to define the name of the executable mystring

$(CFLAGS) is used to define the intended flags for the compilation "-g -Wall -Wshadow"

Makefile

```
GCC = gcc
CFLAGS = -g -Wall -Wshadow
OBJS = mystring.o main.o
HDRS = mystring.h
VAL = valgrind --tool=memcheck --leak-check=full
VAL += --verbose --log-file=


%.o: %.c
	$(GCC) $(CFLAGS) -c $*.c

mystring: $(OBJS) $(HDRS)
	$(GCC) $(CFLAGS) $(OBJS) -o $@

clean:
	rm -f mystring $(OBJS)
```

```
~/Documents/courses/cs2263/lecture/lecture9/L9src/mystring $ cat ou
length: 21
count(t): 2
THIS IS A GREAT IDEA
```

```
~/Documents/courses/cs2263/lecture/lecture9/L9src/mystring $ make
gcc -g -Wall -Wshadow -c main.c
gcc -g -Wall -Wshadow mystring.o main.o -o mystring
```

```
int main(int argc, char *argv[]){
   if (argc != 4) {
      printf("usage: %s command input output\n", argv[0]);
      return EXIT_FAILURE;
   }

   FILE *infptr = fopen(argv[2], "r");
   if (infptr == NULL) {
      printf("unable to open file %s!\n", argv[2]);
      return EXIT_FAILURE;
   }
   FILE *outfptr = fopen(argv[3], "w");
   if (outfptr == NULL) {
      printf("unable to open file %s!\n", argv[3]);
      fclose(infptr);
```

```
    return EXIT_FAILURE;
}
int num_lines = 0;
char buffer[LINE_SIZE];

// count the number of lines in the file
while (fgets(buffer, LINE_SIZE, infptr) != NULL)
    num_lines++;

// return to the beginning of the file
fseek(infptr, 0, SEEK_SET);

char **lines = malloc(sizeof(char *) * num_lines);
int i;
for (i = 0; i < num_lines; i++) {
    if (feof(infptr)){
        printf("not enough num_lines in file!\n");
        fclose(infptr);
        fclose(outfptr);
        return EXIT_FAILURE;
    }
    lines[i] = malloc(sizeof(char) * LINE_SIZE);
    fgets(lines[i], LINE_SIZE, infptr);
}
fclose(infptr);
int total_length = 0;
for (i = 0; i < num_lines; i++)
    total_length += my_strlen(lines[i]);

// count the length of each line
    for (i = 0; i < num_lines; i++){
        fprintf(outfptr, "length: %d\n",
        my_strlen(lines[i]));
    }
/* for each line, count the occurrence of the first
letter in the line */
    for (i = 0; i < num_lines; i++){
        fprintf(outfptr, "count(%c): %d\n", lines[i][0],
        my_countchar(lines[i], lines[i][0]));
    }
    for (i = 0; i < num_lines; i++) {
        my_strupper(lines[i]);
        fprintf(outfptr, "%s", lines[i]);
    }
   for (i = 0; i < num_lines; i++){
        char* ret = my_strchr(lines[i], 'a');
```

```
            fprintf(outfptr, "first a located at: %s\n", ret);
                                                }
    for (i = 0; i < num_lines; i++)
        free(lines[i]);
    free(lines);
    fclose(outfptr);
    return EXIT_SUCCESS;
}
```