```
Yang Zheng
CS347 Spring23
Assignment3
```

```
/* CSCI347 Spring23
 * Assignment 2
 * Modified April 11, 2023 Yang zheng
 */

#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <errno.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdbool.h>
#include "defn.h"

/* Constants */

#define LINELEN 1024
int args = 0;
int shift = 0;
int arg_count = 0;
char** command_line = NULL;

/* Prototypes */

void processline (char *line);

void off_quote(char *line) {
  int j = 0;
  int lineLength = strlen(line);
  for (int i = 0; i < lineLength; i++) {
    if (line[i] != '"') {
        line[j++] = line[i];
    }
  }
  line[j] = '\0';
}

/* find the comment and get rid of the comment */
void off_comment(char *line) {
  char* start = line;
  while (*start != '\0') {
    if (*start == '#' && *(start - 1) != '$') {
      *start = '\0';
      break;
    }
    start++;
  }
}

bool is_empty_or_spaces(char *line) {
    int i = 0;
    while (line[i] != '\0') {
        if (line[i] != ' ' && line[i] != '\t' && line[i] != '\n') {
            return false;  // found non-space character, line is not empty or fu
ll of spaces
        }
        i++;
    }
    return true;  // end of line reached without finding non-space character, li
ne is empty or full of spaces
}

char** arg_parse (char *line, int *argcptr) {
  int count = 1;
  int i = 0;
  bool no_quote = true;
  int length = strlen(line);

  while (line[i] != 0 && i < length) {
    if (line[i] != ’ ’) {
      while (line[i] != 0 && i < length) {
              if (line[i] == ’"’) {
          no_quote = !no_quote;
        }
        if (line[i] == ’ ’) {
          if (no_quote == false) { // if we have read a \", don’t do anything
            ;
          } else {
            count++;
            break;
          }
        }
        i++;
      }
      i++;
    } else {
      i++;
    }
  }

  if (no_quote == false) {
    fprintf(stderr, "No matching double quotes");
  }

  i = 0;
  int j = 0;
  char** arr = (char**) malloc ((count + 1) * sizeof(char*));
  if (arr == NULL) {
     fprintf (stderr, "Failed to malloc");
  }

  while (line[i] != 0 && i < length) {
    if (line[i] != ’ ’) {
      arr[j] = &line[i];
      j++;
      while (line[i] != 0 && i < length) {
              if (line[i] == ’"’) {
          no_quote = !no_quote;
        }
        if (line[i] == ’ ’) {
          if (no_quote == false) { // if we have read a \", don’t do anything
            ;
          } else {
            line[i] = 0;
            break;
          }
        }
        i++;
      }
      i++;
    } else {
      i++;
    }
  }

  for (int i = 0; i < j; i++) {
    off_quote(arr[i]);
  }

  arr[count] = NULL;
  *argcptr = count;

  // for (int i = 0; i < count; i++) {
  //   printf("arr[%d]: %s\n", i, arr[i]);
  // }
  return arr;
}


/* Shell main */
int
main (int argc, char **argv)
{
  // for (int i = 0; i < argc; i++) {
  //   printf("argv[%d]: %s\n", i, argv[i]);
  // }
  arg_count = argc - 1;
  args = argc - 1; //  args starts from index 2 to index n - 1 of the command line
  command_line = argv;
  char buffer[LINELEN];
  int len;
  FILE* read;
  if (argc == 1) {
    read = stdin;
  } else {
    // char* filename = argv[1];
    read = fopen(argv[1], "r");
    if (read == NULL) {
      fprintf(stderr, "Failed to open file %s\n", argv[1]);
      exit(127);
    }
  }
  while (1) {

    /* prompt and get line */
    if (read == stdin) {
      fprintf (stderr, "%% ");
    }

    if (fgets (buffer, LINELEN, read) != buffer) {
      break;
    }

    if (!is_empty_or_spaces(buffer)) {
      /* Get rid of \n at end of buffer. */
      // printf("buffer: %s\n", buffer);
      len = strlen(buffer);
      if (buffer[len-1] == ’\n’)
          buffer[len-1] = 0;
      off_comment(buffer);
      /* Run it ... */
      processline (buffer);
    }
    if (feof(read)) {
      break;
    }
  }

  if (!feof(read)) {
    perror ("read");
  }

  fclose(read);
  return 0;             /* Also known as exit (0); */
}


void processline (char *line)
{
    pid_t  cpid;
    int    status;

    char newLine[LINELEN] = {0};
    int condition = expand(line, newLine, LINELEN);
    // printf("newLine: %s\n", newLine);
    if (condition == -1) { // if expand failed, print error message
      fprintf(stderr, "Expand failed\n");
      return;
    }

    int argc = 0;
    char** p_arr = arg_parse(newLine, &argc);
    // printf("p_arr[0]: %s\n", p_arr[0]);
    if (newLine == NULL || p_arr[0] == NULL) {
      return;
    }

    /* check if new line contains builtin command before fork */
    if (exec_builtin(p_arr) < 0) {
      /* Start a new process to do the job. */
      cpid = fork();
      if (cpid < 0) {
        /* Fork wasn’t successful */
        perror ("fork");
        return;
      }

      /* Check for who we are! */
      if (cpid == 0) {
        /* We are the child! */
        // printf("p_arr[0]: %s\n", p_arr[0]);
        execvp(p_arr[0], p_arr);

        /* execlp reurned, wasn’t successful */
        perror ("exec");
        fclose(stdin);  // avoid a linux stdio bug
        exit (127);
      }

      /* free pointer array */
      free(p_arr);
      p_arr = NULL;

      /* Have the parent wait for child to complete */
      if (wait (&status) < 0) {
        /* Wait wasn't successful */
        perror ("wait");
      }
    } else {
      // free(p_arr);
      // p_arr = NULL;
      ;
    }

}
```

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#include <stdlib.h>
#include <unistd.h>
#include <ctype.h>
#include <dirent.h>
#include "defn.h"

int result = 0; //  result of expand

void cat(char* new, char* to_cat, int* space) {
    // printf("space: %d, to_cat: %d, new: %d\n", *space, strlen(to_cat), strlen
(new));
    if (strlen(to_cat) + strlen(new) <= *space) {
        strcat(new, to_cat);
        *space -= strlen(to_cat);
    } else {
        fprintf(stderr, "No enough space to add\n");
    }
}

int expand (char *orig, char *new, int newsize) {
    // need a pointer points to the first char of NAME
    char *name = orig;

    // another pointer finds the first '}' and set it to '\0'
    char *end = orig;
    char* value = 0; // the value of the environment variable
    char pid_str[16] = {0};
    int space = newsize;
    bool has_quote = false; //  if we read a ${, we set it to true
    // printf("orig: %s\n", orig);
    if (orig[5] == '}') {
        printf("fuck\n");
    }
    while (*name != '\0' && *end != '\0') {
        while (*name != '{') {
            if (*name == '\0') { //  if we never read a {
                if (new[strlen(new) - 1] == ' ') {
                    new[strlen(new) - 1] = '\0';
                }
                return result;
            }
            if (*name == '$'){
                name++;
                if (*name == '$') { //  this will increment name
                    if (sprintf(pid_str, "%d", getpid()) >= 0) {
                        cat(new, pid_str, &space);
                    } else {
                        fprintf(stderr, "failed to get pid");
                        result = -1;
                        return result;
                    }
                } else if (*name == '{') {
                    has_quote = !has_quote;
                    break;
                } else if (isdigit(*name)) {
                    char num[10] = {0};
                    if (args > 0) {
                        while (isdigit(*name)) {
                            char n = *name;
                            strcat(num, &n);
                            name++;
                        }
                        int pattern_n = atoi(num);
                        if (pattern_n >= args) {
                            cat(new, "", &space);
                        } else {
                            cat(new, command_line[pattern_n + 1 + shift], &space
); //  out of bounds?
                        }
                        name--;
                    } else { // interactive mode
                        if (atoi(num) == 0) {
                            cat(new, "./ush", &space);
                        } else {
                            cat(new, "", &space);
                        }
                    }
                } else if (*name == ’#’) {
                    char pound[3] = {0};
                    if (args > 0) {
                        if (sprintf(pound, "%d", args) >= 0) {
                            cat(new, pound, &space);
                        } else {
                            fprintf(stderr, "failed to get #");
                            result = -1;
                            return result;
                        }
                    } else {
                        cat(new, "1", &space);
                    }
                } else { //  if we read a $ that is not a ${ or $$, we do nothin
g
                    name--;
                    cat(new, name, &space);
                    return result;
                }
            } else if (*name == ’*’) {
                end = (name + 1);
                char* r_express = (name + 1);
                DIR *dir;
                struct dirent *ent;
                dir = opendir(".");
                bool reached_end = false;
                if (*end == ’ ’ || *end == ’\0’) { //  if there is no pattern
                    r_express = "";
                } else {
                    while (*end != ’ ’ && *end != ’\0’) {
                        end++;
                    }
                    if (*end == ’ ’) {
                        *end = ’\0’;
                    } else {
                        reached_end = true;
                    }
                }
                if (dir != NULL) {
                    bool matched = false;
                    if (strchr(r_express, '/') != NULL) {
                            fprintf(stderr, "can't include /\n");
                            result = -1;
                            return result;
                    }
                    while ((ent = readdir(dir)) != NULL) {
                        if (strcmp(ent->d_name + strlen(ent->d_name) - strlen(r_
express), r_express) == 0
                        && ent->d_name[0] != '.') {
                            matched = true;
                            cat(new, ent->d_name, &space);
                            cat(new, " ", &space);
                        }
                    }
                    if (matched == false) { //  if we can't find matching files
                        cat(new, r_express, &space);
                    }
                    closedir(dir);
                } else {
                    perror("Failed to open directory");
                    result = -1;
                    return result;
                }
                if (reached_end) {
                    if (new[strlen(new) - 1] == ' ') {
                        new[strlen(new) - 1] = '\0';
                    }
                    // printf("here\n");
                    break;
                } else {
                    name = end;
                    *end = ' ';

                }
            } else if (*name == '\\') {
                if (*(name + 1) == '*') {
                    cat(new, "*", &space);
                }
                while (*name != ' ' && *name != '\0') {
                    name++;
                }
                if (*name == '\0') {
                    break;
                }
            } else {
                char append[1] = {0};
                append[0] = orig[name - orig];
                append[1] = '\0';
                cat(new, append, &space);
                if (*name != ' ' && *(name + 1) == '*') {
                    cat(new, "*", &space);
                    name++;
                }

            }
            name++;
```

```c
        }
        // printf("end: %c\n", *end);
        // printf("break\n");
        name++;
        // end = name;
        //set the last char of orig to '\0', now name points to a string
        if (has_quote == true) {
            while (*end != '}') {
                printf("end is at: %ld\n", end - orig);
                if (*end == '\0') {
                    fprintf(stderr, "Error: missing '}'\n");
                    result = -1;
                    return result;
                }
                end++;
            }
            *end = '\0';
            value = getenv(name);
            if (value == NULL) {
                cat(new, "", &space);
            } else {
                cat(new, value, &space);
            }
            *end = '}'; // set it back to '}
            end++;
            name = end;
        }
    }
    result = 1;
    return result;
}
```

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <sys/stat.h>
#include <pwd.h>
#include <grp.h>
#include <time.h>
#include "defn.h"

static char* list[] = {"exit", "envset", "envunset", "cd", "shift", "unshift", "sstat"};
typedef void (*funcPtr) ();
static int is_builtin;
static char** command;

void exec_exit() {
    if (command[1] == NULL) {
        free(command);
        command = NULL;
        exit(0);
    } else {
        int exit_value = atoi(command[1]);
        free(command);
        command = NULL;
        if (exit_value == 0) {
            fprintf(stderr, "not given a valid exit value");
            // is_builtin = -1;
            // return is_builtin;
        }
        exit(exit_value);
    }
}

void exec_envset() {
    char* new_value = command[2];
    int ret = setenv(command[1], new_value, 1);
     if (ret != 0) {
        perror("setenv");
        return;
    }
}

void exec_envunset() {
    if (unsetenv(command[1]) == -1) {
        perror("envunset");
        return;
    }
}

void exec_cd() {
    int result = 0;
    if (command[1] == NULL) {
        result = chdir(getenv("HOME"));
    } else {
        result = chdir(command[1]);
    }
    if (result != 0) {
        perror("chdir");
        // is_builtin = -1;
        // return is_builtin;
    }
}

void exec_shift() {
    if (command[1] == NULL) {
        shift = 1;
    } else {
        shift = atoi(command[1]);
    }
    if ((args - shift) < 0) {
        fprintf(stderr, "can’t shift that many arguments\n");
        // is_builtin = -1;
        // return is_builtin;
    } else {
        args = args - shift;
    }
}

void exec_unshift() {
    if (command[1] != NULL) { //  if we were given the unshift value
        if (atoi(command[1]) > shift) {
            fprintf(stderr, "can’t unshift that many arguments\n");
            // is_builtin = -1;
            // return is_builtin;
        }
        args += atoi(command[1]);
        shift -= atoi(command[1]);
    } else {
        args = arg_count;
        shift = 0;
    }

}
void exec_sstat() {
    char perms[11];
    struct stat st;
    for (int i = 1; i < sizeof(command); i++) {
        if (stat(command[i], &st) == 0) {
            printf("%s ", command[i]); //  print file name

            struct passwd *pwd = getpwuid(st.st_uid);
            if (pwd == NULL) { //  print user name
                printf("%u ", st.st_uid);
            } else {
                printf("%s ", pwd->pw_name);
            }

            struct group *grp = getgrgid(st.st_gid); // print group name
            if (grp == NULL) {
                printf("%u ", st.st_gid);
            } else {
                printf("%s ", grp->gr_name);
            }

            strmode(st.st_mode, perms); //  print permission
            printf("%s ", perms);
            printf("%lu ", st.st_nlink); //  print number of links"
            printf("%lu ", st.st_size); //  print size
            printf("%s\n", asctime(localtime(&st.st_mtime))); //  print last mod
ified time
        }
    }

}

int exec_builtin(char** line) {
    funcPtr flist[] = {exec_exit, exec_envset, exec_envunset, exec_cd, exec_shif
t, exec_unshift, exec_sstat};
    command = line;
    is_builtin = 1;
    for (int i = 0; i < sizeof(list)/sizeof(list[0]); i++) {
        if (strcmp(command[0], list[i]) == 0) {
            flist[i]();
            // is_builtin = 1;
            free(command);
            command = NULL;
            return is_builtin;
        }
    }
    /* didn't find a builtin command */
    is_builtin = -1;
    return is_builtin;
}
```

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

int expand (char *orig, char *new, int newsize);
int exec_builtin(char** line);
void strmode(mode_t mode, char *p);
extern int args;
extern int shift;
extern int arg_count;
extern char** command_line;
```

```
Script started on 2023-05-01 12:19:02-07:00 [TERM="xterm-256color" TTY="/dev/pts
/0" COLUMNS="190" LINES="17"]
^[[?2004h^[]0;zhengy@cf162-07: ~/csci347/csci347_s23/ush^G^[[01;32mzhengy@cf162-
07^[[00m:^[[01;34m~/csci347/csci347_s23/ush^[[00m$ ./ush
^[[?2004l
% echo #this is my own test

% echo *
strmode.c expand.o 3adc test_script a2report.pdf 3.h 4a?c builtin.c a2.pdf scr4.
txt subdir a2_test 2acc d.cc printArg.c~ e.b report.ps~ b.c Makefile ush.o f.q a
1.ps printArg test 1abc own_test fully report.ps a2.ps a1.ps~ a.c c..c a2.ps~ st
rmode.o expand.c printArg.c builtin.o report.pdf aaaaaaa.c script-nq #ush.c#~ us
h ush.c header.txt a2report.ps showshift.txt test.c testa2 defn.h a1.pdf
% echo *.c *.o
strmode.c builtin.c b.c a.c c..c expand.c printArg.c aaaaaaa.c ush.c test.c expa
nd.o ush.o strmode.o builtin.o
% sstat showshift.txt
showshift.txt zhengy grp.csci.Students -rw-r--r--  1 414 Sun Apr 30 23:39:57 202
3

% echo a*
a*
% eho^H ^H^H ^Hcho c*
c*
% echo \*
*
% echo ?^H ^H/*
/*
% echo */
can't include /
Expand failed
% ss^H ^H^H ^H^C
^[[?2004h^[]0;zhengy@cf162-07: ~/csci347/csci347_s23/ush^G^[[01;32mzhengy@cf162-
07^[[00m:^[[01;34m~/csci347/csci347_s23/ush^[[00m$ ./ush showshift.txt a b c d
^H^[[Ke f
^[[?2004l
showshift is named showshift.txt
Number of arguments is 7.
Argument 1 is a.
Argument 2 is b.
Argument 3 is c.
Argument 4 is d.
Number of arguments is 4.
Argument 1 is d.
Argument 2 is e.
Argument 3 is f.
Argument 4 is .
Number of arguments is 5.
Argument 1 is c.
Number of arguments is 7.
Now a is Argument 1.
^[[?2004h^[]0;zhengy@cf162-07: ~/csci347/csci347_s23/ush^G^[[01;32mzhengy@cf162-
07^[[00m:^[[01;34m~/csci347/csci347_s23/ush^[[00m$ exit
^[[?2004l
exit

Script done on 2023-05-01 12:20:54-07:00 [COMMAND_EXIT_CODE="0"]
Script started on 2023-05-01 12:21:08-07:00 [TERM="xterm-256color" TTY="/dev/pts
/0" COLUMNS="190" LINES="17"]
^[[?2004h^[]0;zhengy@cf162-07: ~/csci347/csci347_s23/ush^G^[[01;32mzhengy@cf162-
07^[[00m:^[[01;34m~/csci347/csci347_s23/ush^[[00m$ ./ius^H^[[K^H^[[K^H^[[K^H^[[K
ush^H^[[K^H^[[K^H^[[K/ush
^[[?2004l
% cd  /home/phil/public/csci347/testa2^H3
% ./try -H
mkdir: cannot create directory âM-^@M-^X/home/zhengy/347_test_a3âM-^@M-^Y: File
exists
~/347_test_a3 exists, use it anyway? (y/n) y
Cloning into 'csci347_s23'...
remote: Enumerating objects: 269, done.^[[K
remote: Counting objects:   0% (1/245)^[[K
remote: Counting objects:   1% (3/245)^[[K
remote: Counting objects:   2% (5/245)^[[K
remote: Counting objects:   3% (8/245)^[[K
remote: Counting objects:   4% (10/245)^[[K
remote: Counting objects:   5% (13/245)^[[K
remote: Counting objects:   6% (15/245)^[[K
remote: Counting objects:   7% (18/245)^[[K
remote: Counting objects:   8% (20/245)^[[K
remote: Counting objects:   9% (23/245)^[[K
remote: Counting objects:  10% (25/245)^[[K
remote: Counting objects:  11% (27/245)^[[K
remote: Counting objects:  12% (30/245)^[[K
remote: Counting objects:  13% (32/245)^[[K
remote: Counting objects:  14% (35/245)^[[K
remote: Counting objects:  15% (37/245)^[[K
remote: Counting objects:  16% (40/245)^[[K
remote: Counting objects:  17% (42/245)^[[K
remote: Counting objects:  18% (45/245)^[[K
remote: Counting objects:  19% (47/245)^[[K
remote: Counting objects:  20% (49/245)^[[K
remote: Counting objects:  21% (52/245)^[[K
remote: Counting objects:  22% (54/245)^[[K
remote: Counting objects:  23% (57/245)^[[K
remote: Counting objects:  24% (59/245)^[[K
remote: Counting objects:  25% (62/245)^[[K
remote: Counting objects:  26% (64/245)^[[K
remote: Counting objects:  27% (67/245)^[[K
remote: Counting objects:  28% (69/245)^[[K
remote: Counting objects:  29% (72/245)^[[K
remote: Counting objects:  30% (74/245)^[[K
remote: Counting objects:  31% (76/245)^[[K
remote: Counting objects:  32% (79/245)^[[K
remote: Counting objects:  33% (81/245)^[[K
remote: Counting objects:  34% (84/245)^[[K
remote: Counting objects:  35% (86/245)^[[K
remote: Counting objects:  36% (89/245)^[[K
remote: Counting objects:  37% (91/245)^[[K
remote: Counting objects:  38% (94/245)^[[K
remote: Counting objects:  39% (96/245)^[[K
remote: Counting objects:  40% (98/245)^[[K
remote: Counting objects:  41% (101/245)^[[K
remote: Counting objects:  42% (103/245)^[[K
remote: Counting objects:  43% (106/245)^[[K
remote: Counting objects:  44% (108/245)^[[K
remote: Counting objects:  45% (111/245)^[[K
```

```
remote: Counting objects:  46% (113/245)^[[K
remote: Counting objects:  47% (116/245)^[[K
remote: Counting objects:  48% (118/245)^[[K
remote: Counting objects:  49% (121/245)^[[K
remote: Counting objects:  50% (123/245)^[[K
remote: Counting objects:  51% (125/245)^[[K
remote: Counting objects:  52% (128/245)^[[K
remote: Counting objects:  53% (130/245)^[[K
remote: Counting objects:  54% (133/245)^[[K
remote: Counting objects:  55% (135/245)^[[K
remote: Counting objects:  56% (138/245)^[[K
remote: Counting objects:  57% (140/245)^[[K
remote: Counting objects:  58% (143/245)^[[K
remote: Counting objects:  59% (145/245)^[[K
remote: Counting objects:  60% (147/245)^[[K
remote: Counting objects:  61% (150/245)^[[K
remote: Counting objects:  62% (152/245)^[[K
remote: Counting objects:  63% (155/245)^[[K
remote: Counting objects:  64% (157/245)^[[K
remote: Counting objects:  65% (160/245)^[[K
remote: Counting objects:  66% (162/245)^[[K
remote: Counting objects:  67% (165/245)^[[K
remote: Counting objects:  68% (167/245)^[[K
remote: Counting objects:  69% (170/245)^[[K
remote: Counting objects:  70% (172/245)^[[K
remote: Counting objects:  71% (174/245)^[[K
remote: Counting objects:  72% (177/245)^[[K
remote: Counting objects:  73% (179/245)^[[K
remote: Counting objects:  74% (182/245)^[[K
remote: Counting objects:  75% (184/245)^[[K
remote: Counting objects:  76% (187/245)^[[K
remote: Counting objects:  77% (189/245)^[[K
remote: Counting objects:  78% (192/245)^[[K
remote: Counting objects:  79% (194/245)^[[K
remote: Counting objects:  80% (196/245)^[[K
remote: Counting objects:  81% (199/245)^[[K
remote: Counting objects:  82% (201/245)^[[K
remote: Counting objects:  83% (204/245)^[[K
remote: Counting objects:  84% (206/245)^[[K
remote: Counting objects:  85% (209/245)^[[K
remote: Counting objects:  86% (211/245)^[[K
remote: Counting objects:  87% (214/245)^[[K
remote: Counting objects:  88% (216/245)^[[K
remote: Counting objects:  89% (219/245)^[[K
remote: Counting objects:  90% (221/245)^[[K
remote: Counting objects:  91% (223/245)^[[K
remote: Counting objects:  92% (226/245)^[[K
remote: Counting objects:  93% (228/245)^[[K
remote: Counting objects:  94% (231/245)^[[K
remote: Counting objects:  95% (233/245)^[[K
remote: Counting objects:  96% (236/245)^[[K
remote: Counting objects:  97% (238/245)^[[K
remote: Counting objects:  98% (241/245)^[[K
remote: Counting objects:  99% (243/245)^[[K
remote: Counting objects: 100% (245/245)^[[K
remote: Counting objects: 100% (245/245), done.^[[K
remote: Compressing objects:   0% (1/241)^[[K
remote: Compressing objects:   1% (3/241)^[[K
remote: Compressing objects:   2% (5/241)^[[K
```

```
remote: Compressing objects:   3% (8/241)^[[K
remote: Compressing objects:   4% (10/241)^[[K
remote: Compressing objects:   5% (13/241)^[[K
remote: Compressing objects:   6% (15/241)^[[K
remote: Compressing objects:   7% (17/241)^[[K
remote: Compressing objects:   8% (20/241)^[[K
remote: Compressing objects:   9% (22/241)^[[K
remote: Compressing objects:  10% (25/241)^[[K
remote: Compressing objects:  11% (27/241)^[[K
remote: Compressing objects:  12% (29/241)^[[K
remote: Compressing objects:  13% (32/241)^[[K
remote: Compressing objects:  14% (34/241)^[[K
remote: Compressing objects:  15% (37/241)^[[K
remote: Compressing objects:  16% (39/241)^[[K
remote: Compressing objects:  17% (41/241)^[[K
remote: Compressing objects:  18% (44/241)^[[K
remote: Compressing objects:  19% (46/241)^[[K
remote: Compressing objects:  20% (49/241)^[[K
remote: Compressing objects:  21% (51/241)^[[K
remote: Compressing objects:  22% (54/241)^[[K
remote: Compressing objects:  23% (56/241)^[[K
remote: Compressing objects:  24% (58/241)^[[K
remote: Compressing objects:  25% (61/241)^[[K
remote: Compressing objects:  26% (63/241)^[[K
remote: Compressing objects:  27% (66/241)^[[K
remote: Compressing objects:  28% (68/241)^[[K
remote: Compressing objects:  29% (70/241)^[[K
remote: Compressing objects:  30% (73/241)^[[K
remote: Compressing objects:  31% (75/241)^[[K
remote: Compressing objects:  32% (78/241)^[[K
remote: Compressing objects:  33% (80/241)^[[K
remote: Compressing objects:  34% (82/241)^[[K
remote: Compressing objects:  35% (85/241)^[[K
remote: Compressing objects:  36% (87/241)^[[K
remote: Compressing objects:  37% (90/241)^[[K
remote: Compressing objects:  38% (92/241)^[[K
remote: Compressing objects:  39% (94/241)^[[K
remote: Compressing objects:  40% (97/241)^[[K
remote: Compressing objects:  41% (99/241)^[[K
remote: Compressing objects:  42% (102/241)^[[K
remote: Compressing objects:  43% (104/241)^[[K
remote: Compressing objects:  44% (107/241)^[[K
remote: Compressing objects:  45% (109/241)^[[K
remote: Compressing objects:  46% (111/241)^[[K
remote: Compressing objects:  47% (114/241)^[[K
remote: Compressing objects:  48% (116/241)^[[K
remote: Compressing objects:  49% (119/241)^[[K
remote: Compressing objects:  50% (121/241)^[[K
remote: Compressing objects:  51% (123/241)^[[K
remote: Compressing objects:  52% (126/241)^[[K
remote: Compressing objects:  53% (128/241)^[[K
remote: Compressing objects:  54% (131/241)^[[K
remote: Compressing objects:  55% (133/241)^[[K
remote: Compressing objects:  56% (135/241)^[[K
remote: Compressing objects:  57% (138/241)^[[K
remote: Compressing objects:  58% (140/241)^[[K
remote: Compressing objects:  59% (143/241)^[[K
remote: Compressing objects:  60% (145/241)^[[K
remote: Compressing objects:  61% (148/241)^[[K
```

```
remote: Compressing objects:  62% (150/241)^[[K
remote: Compressing objects:  63% (152/241)^[[K
remote: Compressing objects:  64% (155/241)^[[K
remote: Compressing objects:  65% (157/241)^[[K
remote: Compressing objects:  66% (160/241)^[[K
remote: Compressing objects:  67% (162/241)^[[K
remote: Compressing objects:  68% (164/241)^[[K
remote: Compressing objects:  69% (167/241)^[[K
remote: Compressing objects:  70% (169/241)^[[K
remote: Compressing objects:  71% (172/241)^[[K
remote: Compressing objects:  72% (174/241)^[[K
remote: Compressing objects:  73% (176/241)^[[K
remote: Compressing objects:  74% (179/241)^[[K
remote: Compressing objects:  75% (181/241)^[[K
remote: Compressing objects:  76% (184/241)^[[K
remote: Compressing objects:  77% (186/241)^[[K
remote: Compressing objects:  78% (188/241)^[[K
remote: Compressing objects:  79% (191/241)^[[K
remote: Compressing objects:  80% (193/241)^[[K
remote: Compressing objects:  81% (196/241)^[[K
remote: Compressing objects:  82% (198/241)^[[K
remote: Compressing objects:  83% (201/241)^[[K
remote: Compressing objects:  84% (203/241)^[[K
remote: Compressing objects:  85% (205/241)^[[K
remote: Compressing objects:  86% (208/241)^[[K
remote: Compressing objects:  87% (210/241)^[[K
remote: Compressing objects:  88% (213/241)^[[K
remote: Compressing objects:  89% (215/241)^[[K
remote: Compressing objects:  90% (217/241)^[[K
remote: Compressing objects:  91% (220/241)^[[K
remote: Compressing objects:  92% (222/241)^[[K
remote: Compressing objects:  93% (225/241)^[[K
remote: Compressing objects:  94% (227/241)^[[K
remote: Compressing objects:  95% (229/241)^[[K
remote: Compressing objects:  96% (232/241)^[[K
remote: Compressing objects:  97% (234/241)^[[K
remote: Compressing objects:  98% (237/241)^[[K
remote: Compressing objects:  99% (239/241)^[[K
remote: Compressing objects: 100% (241/241)^[[K
remote: Compressing objects: 100% (241/241), done.^[[K
Receiving objects:   0% (1/269)
Receiving objects:   1% (3/269)
Receiving objects:   2% (6/269)
Receiving objects:   3% (9/269)
Receiving objects:   4% (11/269)
Receiving objects:   5% (14/269)
Receiving objects:   6% (17/269)
Receiving objects:   7% (19/269)
Receiving objects:   8% (22/269)
Receiving objects:   9% (25/269)
Receiving objects:  10% (27/269)
Receiving objects:  11% (30/269)
Receiving objects:  12% (33/269)
Receiving objects:  13% (35/269)
Receiving objects:  14% (38/269)
Receiving objects:  15% (41/269)
Receiving objects:  16% (44/269)
Receiving objects:  17% (46/269)
Receiving objects:  18% (49/269)
Receiving objects:  19% (52/269)
Receiving objects:  20% (54/269)
Receiving objects:  21% (57/269)
Receiving objects:  22% (60/269)
Receiving objects:  23% (62/269)
Receiving objects:  24% (65/269)
Receiving objects:  25% (68/269)
Receiving objects:  26% (70/269)
Receiving objects:  27% (73/269)
Receiving objects:  28% (76/269)
Receiving objects:  29% (79/269)
Receiving objects:  30% (81/269)
Receiving objects:  31% (84/269)
Receiving objects:  32% (87/269)
Receiving objects:  33% (89/269)
Receiving objects:  34% (92/269)
Receiving objects:  35% (95/269)
Receiving objects:  36% (97/269)
Receiving objects:  37% (100/269)
Receiving objects:  38% (103/269)
Receiving objects:  39% (105/269)
Receiving objects:  40% (108/269)
Receiving objects:  41% (111/269)
Receiving objects:  42% (113/269)
Receiving objects:  43% (116/269)
Receiving objects:  44% (119/269)
Receiving objects:  45% (122/269)
Receiving objects:  46% (124/269)
Receiving objects:  47% (127/269)
Receiving objects:  48% (130/269)
Receiving objects:  49% (132/269)
Receiving objects:  50% (135/269)
Receiving objects:  51% (138/269)
Receiving objects:  52% (140/269)
Receiving objects:  53% (143/269)
Receiving objects:  54% (146/269)
Receiving objects:  55% (148/269)
Receiving objects:  56% (151/269)
Receiving objects:  57% (154/269)
Receiving objects:  58% (157/269)
Receiving objects:  59% (159/269)
Receiving objects:  60% (162/269)
Receiving objects:  61% (165/269)
Receiving objects:  62% (167/269)
Receiving objects:  63% (170/269)
Receiving objects:  64% (173/269)
Receiving objects:  65% (175/269)
Receiving objects:  66% (178/269)
Receiving objects:  67% (181/269)
Receiving objects:  68% (183/269)
Receiving objects:  69% (186/269)
Receiving objects:  70% (189/269)
Receiving objects:  71% (191/269)
Receiving objects:  72% (194/269)
Receiving objects:  73% (197/269)
Receiving objects:  74% (200/269)
Receiving objects:  75% (202/269)
Receiving objects:  76% (205/269)
Receiving objects:  77% (208/269)
Receiving objects:  78% (210/269)
Receiving objects:  79% (213/269)
Receiving objects:  80% (216/269)
Receiving objects:  81% (218/269)
Receiving objects:  82% (221/269)
Receiving objects:  83% (224/269)
Receiving objects:  84% (226/269)
Receiving objects:  85% (229/269)
Receiving objects:  86% (232/269)
Receiving objects:  87% (235/269)
Receiving objects:  88% (237/269)
Receiving objects:  89% (240/269)
Receiving objects:  90% (243/269)
Receiving objects:  91% (245/269)
Receiving objects:  92% (248/269)
Receiving objects:  93% (251/269)
Receiving objects:  94% (253/269)
Receiving objects:  95% (256/269)
remote: Total 269 (delta 147), reused 0 (delta 0), pack-reused 24^[[K
Receiving objects:  96% (259/269)
Receiving objects:  97% (261/269)
Receiving objects:  98% (264/269)
Receiving objects:  99% (267/269)
Receiving objects: 100% (269/269)
Receiving objects: 100% (269/269), 244.23 KiB | 2.87 MiB/s, done.
Resolving deltas:   0% (0/153)
Resolving deltas:   1% (2/153)
Resolving deltas:   2% (4/153)
Resolving deltas:   3% (5/153)
Resolving deltas:   4% (7/153)
Resolving deltas:   5% (8/153)
Resolving deltas:   6% (10/153)
Resolving deltas:   7% (11/153)
Resolving deltas:   8% (13/153)
Resolving deltas:   9% (14/153)
Resolving deltas:  10% (16/153)
Resolving deltas:  11% (17/153)
Resolving deltas:  12% (19/153)
Resolving deltas:  13% (20/153)
Resolving deltas:  14% (22/153)
Resolving deltas:  15% (23/153)
Resolving deltas:  16% (25/153)
Resolving deltas:  17% (27/153)
Resolving deltas:  18% (28/153)
Resolving deltas:  19% (30/153)
Resolving deltas:  20% (31/153)
Resolving deltas:  21% (33/153)
Resolving deltas:  22% (34/153)
Resolving deltas:  23% (36/153)
Resolving deltas:  24% (37/153)
Resolving deltas:  25% (39/153)
Resolving deltas:  26% (40/153)
Resolving deltas:  27% (42/153)
Resolving deltas:  28% (43/153)
Resolving deltas:  29% (45/153)
Resolving deltas:  30% (46/153)
Resolving deltas:  31% (48/153)
Resolving deltas:  32% (49/153)
Resolving deltas:  33% (51/153)
Resolving deltas:  34% (53/153)
Resolving deltas:  35% (54/153)
Resolving deltas:  36% (56/153)
Resolving deltas:  37% (57/153)
Resolving deltas:  38% (59/153)
Resolving deltas:  39% (60/153)
Resolving deltas:  40% (62/153)
Resolving deltas:  41% (63/153)
Resolving deltas:  42% (65/153)
Resolving deltas:  43% (66/153)
Resolving deltas:  44% (68/153)
Resolving deltas:  45% (69/153)
Resolving deltas:  46% (71/153)
Resolving deltas:  47% (72/153)
Resolving deltas:  48% (74/153)
Resolving deltas:  49% (75/153)
Resolving deltas:  50% (77/153)
Resolving deltas:  51% (79/153)
Resolving deltas:  52% (81/153)
Resolving deltas:  53% (82/153)
Resolving deltas:  54% (83/153)
Resolving deltas:  55% (85/153)
Resolving deltas:  56% (86/153)
Resolving deltas:  57% (88/153)
Resolving deltas:  58% (89/153)
Resolving deltas:  59% (91/153)
Resolving deltas:  60% (92/153)
Resolving deltas:  61% (94/153)
Resolving deltas:  62% (95/153)
Resolving deltas:  63% (97/153)
Resolving deltas:  64% (98/153)
Resolving deltas:  65% (100/153)
Resolving deltas:  66% (101/153)
Resolving deltas:  67% (103/153)
Resolving deltas:  68% (105/153)
Resolving deltas:  69% (106/153)
Resolving deltas:  70% (108/153)
Resolving deltas:  71% (109/153)
Resolving deltas:  72% (111/153)
Resolving deltas:  73% (112/153)
Resolving deltas:  74% (114/153)
Resolving deltas:  75% (115/153)
Resolving deltas:  76% (117/153)
Resolving deltas:  77% (118/153)
Resolving deltas:  78% (120/153)
Resolving deltas:  79% (121/153)
Resolving deltas:  80% (123/153)
Resolving deltas:  81% (124/153)
Resolving deltas:  82% (126/153)
Resolving deltas:  83% (127/153)
Resolving deltas:  84% (129/153)
Resolving deltas:  85% (131/153)
Resolving deltas:  86% (132/153)
Resolving deltas:  87% (134/153)
Resolving deltas:  88% (135/153)
Resolving deltas:  89% (137/153)
Resolving deltas:  90% (138/153)
Resolving deltas:  91% (140/153)
Resolving deltas:  92% (141/153)
Resolving deltas:  93% (143/153)
Resolving deltas:  94% (144/153)
Resolving deltas:  95% (146/153)
Resolving deltas:  96% (147/153)
Resolving deltas:  97% (149/153)
Resolving deltas:  98% (150/153)
Resolving deltas:  99% (152/153)
Resolving deltas: 100% (153/153)
Resolving deltas: 100% (153/153), done.
Branch 'a3' set up to track remote branch 'a3' from 'origin'.
Switched to a new branch 'a3'
Making REF
gcc -g -Wall -c builtin.c
gcc -g -Wall -c strmode.c
gcc -g -Wall -o ush ush.o expand.o builtin.o strmode.o

run tests? y
Running ush

./try: line 189: 647121 Segmentation fault      ./ush < ${TOP}/ush.test > OUT 2>
 ERRS
Files REF and OUT differ
Script output different!
Diffs:
--- REF 2023-05-01 12:22:14.045020643 -0700
+++ OUT 2023-05-01 12:22:16.469024182 -0700
@@ -1,56 +1,2 @@

 This test script is run with 1 args, script is ./ush, should be interactive.
-Testing args!
-Script name is /home/phil/public/csci347/testa3/scr1.
-There are 4 number of parameters.
-Dollar 1 is 'with', 2 is 'some'.
-'/home/phil/public/csci347/testa3/scr1' is the script name.
-Testing args!
-Script name is /home/phil/public/csci347/testa3/scr1.
-There are 2 number of parameters.
-Dollar 1 is 'with_one_arg', 2 is ''.
-'/home/phil/public/csci347/testa3/scr1' is the script name.
-Testing shift ...
-Script name is /home/phil/public/csci347/testa3/scr2.
-There are 11 number of parameters.
-Dollar 1-4 is arg1, arg2, arg3, and arg4.
-Shifting 1
-There are 10 number of parameters.
-Dollar 1-4 is arg2, arg3, arg4, and arg5.
-Shifting 2
-There are 8 number of parameters.
-Dollar 1-4 is arg4, arg5, arg6, and arg7.
-Shifting 4
-There are 4 number of parameters.
-There are 5 number of parameters.
-Dollar 1-2 is arg7 and arg8.
-There are 7 number of parameters.
-Dollar 1-4 is arg5, arg6, arg7, and arg8.
-There are 11 number of parameters.
-Dollar 1-4 is arg1, arg2, arg3, and arg4.
-There are 21 arguments.
-Argument 13 is 'This_is_thirteen!'.
```

```
-Testing Wildcards and sstat ...
-
-1abc 2acc 3.h 3adc 4a?c a.c aaaaaaa.c b.c c..c d.cc e.b f.q
-d.cc
-a*
-*a
-1abc 2acc 3adc 4a?c a.c aaaaaaa.c b.c c..c d.cc
-c..c
-3.h
-aaaaaaa.c
-*.c
-4a?c
-a.c aaaaaaa.c
-
-Now for sstat
-
-d.cc zhengy grp.csci.Students -rw--w---x 1 0 Sun May  1 00:00:00 2005
-f.q zhengy grp.csci.Students -rw--w---x 1 0 Sun May  1 00:00:00 2005
-/home/phil/zzz.file 333 333 -rwx------ 1 0 Mon Oct 31 15:46:23 2011
-a.c zhengy grp.csci.Students -rw--w---x 1 0 Sun May  1 00:00:00 2005
-b.c zhengy grp.csci.Students -rw--w---x 1 0 Sun May  1 00:00:00 2005
-d.cc zhengy grp.csci.Students -rw--w---x 1 0 Sun May  1 00:00:00 2005
-End testing Wildcards and sstat ...
-Done with regular tests

See Out file? y
Out is: -------

This test script is run with 1 args, script is ./ush, should be interactive.
---------------
Exit values incorrect
Bad exit value on end of script.

----- ERRS -----
% % % %
----------------

Run error tests? y
orig:
orig: echo Testing errors with $0 and $# arguments (should be 5)
p_arr[0]: echo
Testing errors with /home/phil/public/csci347/testa3/ush.err and 5 arguments (sh
ould be 5)
orig: envset N $#
orig: echo Initial arguments are $1, $2, $3, and $4.
p_arr[0]: echo
Initial arguments are arg1, arg2, arg3, and arg4.
orig: echo Doing a shift of 4 (no error)
p_arr[0]: echo
Doing a shift of 4 (no error)
orig: shift 4
orig: echo We now have $# argument, arg1 is ’$1’.
p_arr[0]: echo
We now have 1 argument, arg1 is ’’.
orig: echo Doing a shift of 4 and 1  (should be errors)
p_arr[0]: echo
Doing a shift of 4 and 1 (should be errors)
orig: shift 4
can’t shift that many arguments
```

```
orig: shift
orig: echo Now doing ’unshift 5’ (should be an error)
p_arr[0]: echo
Now doing ’unshift 5’ (should be an error)
orig: unshift 5
can’t unshift that many arguments
orig: unshift
orig: echo Did an unshift ... number of args is $#, should be ${N}.
end is at: 58
p_arr[0]: echo
Did an unshift ... number of args is 5, should be 5.
orig: echo Testing sstat errors
p_arr[0]: echo
Testing sstat errors
orig: sstat notafile
orig: echo End of error tests
p_arr[0]: echo
End of error tests


Look at checked out files? y
^[[?2004h^]0;zhengy@cf162-07: ~/347_test_a3/csci347_s23/ush^G^[[01;32mzhengy@cf
162-07^[[00m:^[[01;34m~/347_test_a3/csci347_s23/ush^[[00m$ exit
^[[?2004l
exit
Clean? y
cleaning
removing ~/347_test_a3
% ^C
^[[?2004h^]0;zhengy@cf162-07: ~/csci347/csci347_s23/ush^G^[[01;32mzhengy@cf162-
07^[[00m:^[[01;34m~/csci347/csci347_s23/ush^[[00m$ exit
^[[?2004l
exit

Script done on 2023-05-01 12:22:42-07:00 [COMMAND_EXIT_CODE="130"]
```