

```
1: /**
2:  * Brandon Rodriguez
3:  * CS 3240
4:  * 09-06-17
5:  * A0 (Assignment 0)
6:  */
7:
8:
9: /**
10:  * Description:
11:  *
12:  * Reintroduction to C, using a SongCSV data file.
13:  *
14:  * First sets program to loop until "ZZZ" command is given.
15:  * Then loads in CSV Song data.
16:  * Finally, it takes in user input and first searches for the corresponding
17:  * command.
18:  * If no command is given, then searches for a song with the given name.
19:  *
20:  * Recognized commands are "All", "Help", and "ZZZ".
21:  */
22:
23:
24: /**
25:  * KNOWN ISSUES:
26:  *
27:  * The "test-input" file provided seems to try searching for a few songs by
28:  * album, not just song name.
29:  *
30:  * However, attempting to implement said search gives "malloc.c systemalloc
31:  * assertion failure". It appears that I have a memory leak somewhere? But
32:  * I don't understand c nearly enough to troubleshoot it.
33:  *
34:  * I even attempted using valgrind, but I don't fully understand what it's
35:  * describing. Funily enough, the error does not seem to occur in valgrind,
36:  * and the album search works fine (if said album_search code is
37:  * uncommented first).
38:  */
39:
40:
41: /**
42:  * Much referencing to
43:  * https://www.tutorialspoint.com/c\_standard\_library/
44:  * For finding documentation on "standard" commands which C seems to make so
45:  * difficult. If it's a command I used in this program, then I probably
46:  * referenced the corresponding tutorialspoint page, at least a little bit.
47:  */
48:
49:
50: /**
51:  * Personal Notes:
52:  *
53:  * Using & before a variable name means "use the memory address of this
54:  * variable".
55:  *
56:  * Char* stringHere; will place the letters (literal string) in read-only memory.
57:  * Char stringHere[]; will place literal string in memory on the stack.
58:  * Thus, in the first, stringHere = 'A' is NOT legal, where it is legal in
59:  * the second.
60:  *
61:  * Typedef essentially gives the provided type a new name.
62:  * IE: You can take any primitive type you want and define it as "Byte",
63:  * "Song", "Cat", or whatever else you may desire. Particularly useful for
```

90
100

some entries aren't found
when searching

```
64:  * structs.
65:  *
66:  * fscanf() is to read user input.
67:  *
68:  * printf() is to print to console.
69:  *
70:  * malloc() allocates the specified amount of memory to the object.
71:  *     Takes "memory size to allocate".
72:  * calloc() does malloc, but sets all values to 0.
73:  *     Takes "number of elements", "individual element size".
74:  * realloc() takes a previous malloc/calloc call and provides more memory.
75:  *     Takes "previous memory pointer", "new memory size".
76:  *
77:  *     All of these return either pointer or null if failed.
78:  *
79:  *
80:  * strncpy() copies string value.
81:  *     Takes "destination", "source", "size".
82:  *
83:  * memcpy() copies memory value.
84:  *     "destination", "source", "size".
85:  *
86:  * strcmp() compares values of two strings.
87:  *     If resulting value < 0, then first string is less than second.
88:  *     If resulting value > 0, then first string is greater than second.
89:  *     If resulting value == 0, then strings are equal.
90:  */
91:
92:
93: // Import headers.
94: #include <stdio.h>
95: #include <string.h>
96: #include <stdlib.h>
97: #include <ctype.h>
98: #include "apue.h"
99:
100:
101: // Define Vars.
102: #define BUFFERSIZE 1024
103: #define ARRAY_SIZE 500;
104:
105: // Variables.
106: typedef struct {
107:     char* artist;
108:     char* song_name;
109:     char* album_name;
110:     float* duration;
111:     int* year;
112:     double* hotttnesss;
113: } songs_struct; // Songs struct.
114:
115: int songs_array_items = ARRAY_SIZE // Saves current size of songs array.
116: songs_struct** songs_array; // The array which holds all song pointers.
117:
118:
119: // Method Declaration.
120: void initialize_data();
121: char* to_lower_case(char* input_string);
122: void buffer_array();
123: char* copy_string(char* dest_ptr, char* source_ptr);
124: int* copy_int(int* dest_ptr, int* source_ptr);
125: float* copy_float(float* dest_ptr, float* source_ptr);
126: double* copy_double(double* dest_ptr, double* source_ptr);
```

Interesting use of pointers
for floats, ints, and doubles... assuming
you particularly want them
on the heap instead of the
stack?

```

127: void sort_array();
128: void find_song(char* song_name);
129: songs_struct* search_array_by_song(int first_index, int last_index, char* desired_recor
d);
130: songs_struct* search_array_by_album(int first_index, int last_index, char* desired_reco
rd);
131: void print_help_text();
132: void print_all_songs();
133: void print_song_info(songs_struct* song);
134:
135:
136: /**
137:  * Program's main.
138:  * Initializes and runs program.
139:  */
140: int main(int argc, char* argv[]) {
141:
142:     // Initialize Variables.
143:     int run_program_bool = 1;    // Bool to keep program running.
144:     char* user_input_string;    // Holds string of user's input.
145:     char* temp_string = calloc(100, sizeof(char));
146:
147:     // Call setup methods.
148:     initialize_data();
149:     print_help_text();
150:
151:     // Main Program Loop.
152:     while (run_program_bool == 1) {
153:
154:         // Get user input.
155:         user_input_string = calloc(100, sizeof(char));
156:         fgets(temp_string, 100, stdin);
157:         strcat(user_input_string, temp_string);
158:         user_input_string = strtok(temp_string, "\n");
159:
160:         // Ensure input is valid and check value.
161:         if (user_input_string != NULL && strcmp(user_input_string, "\n") != 0) {
162:             user_input_string = to_lower_case(user_input_string);
163:
164:
165:             // Check if user needs reprinting of help text.
166:             if (strcmp(user_input_string, "help\0") == 0) {
167:                 print_help_text();
168:             }
169:             // Check if user has requested to exit program.
170:             else if (strcmp(user_input_string, "zzz") == 0) {
171:                 run_program_bool = 0;
172:                 printf("Exiting Program...\n");
173:             }
174:             // Check if user has requested song sorting.
175:             else if (strcmp(user_input_string, "sort") == 0) {
176:                 sort_array();
177:                 printf("Sorted.\n\n");
178:             }
179:             // Check if user has requested to print all songs.
180:             else if (strcmp(user_input_string, "all") == 0) {
181:                 print_all_songs();
182:             }
183:             else { // Attempt to find song with given name.
184:                 char* user_string_with_quotes = calloc(100, sizeof(char));
185:                 strcat(user_string_with_quotes, "\"");
186:                 strcat(user_string_with_quotes, user_input_string);
187:                 strcat(user_string_with_quotes, "\"");

```

```

Main.c          Thu Sep 14 20:32:21 2017          4

188:             find_song(user_string_with_quotes);
189:         }
190:     }
191: }
192:
193:     return 0;
194: }
195:
196:
197: /**
198:  * Initializes program by reading data in from CSV.
199:  */
200: void initialize_data() {
201:     printf("Initializing...\n");
202:
203:     // Initialize variables.
204:     int song_index = 0;                // Holds index of current song.
205:     int song_field_index = 0;          // Holds index of current song field.
206:     char* buffer;                      // Temporary buffer to read in file.
207:     char* token;                       // Temporary token to help parse csv file.
208:     FILE* csv_file;                    // The csv file that is parsed.
209:     char* temp_song;                   // A temporary placeholder for songs.
210:     char* temp_string;                 // A temporary placeholder string.
211:     int temp_int;                      // A temporary placeholder int.
212:     int* temp_int_ptr;                 // A pointer for above int.
213:     double temp_double;                // A temporary placeholder double.
214:     double* temp_double_ptr;           // A pointer for above double.
215:     float temp_float;                  // A temporary placeholder float.
216:     float* temp_float_ptr;             // A pointer for above float.
217:
218:     // Initially allocate memory for variables.
219:     buffer = calloc(1, BUFFERSIZE);
220:     temp_song = calloc(1, BUFFERSIZE);
221:     temp_int = (int) calloc(1, sizeof(int));
222:     songs_array = calloc(songs_array_items, sizeof(songs_struct *));
223:
224:     // Read in data line by line.
225:     printf("Reading csv...\n");
226:     csv_file = fopen("Data/SongCSV.csv", "r");
227:     while (fgets(buffer, BUFFERSIZE, csv_file) != NULL) {
228:
229:         // Check to make sure there are still open spots in songs_array.
230:         if (song_index >= songs_array_items) {
231:             buffer_array();
232:         }
233:
234:         temp_song = strdup(buffer);
235:         song_field_index = 0;
236:         songs_array[song_index] = calloc(1, sizeof(songs_struct));
237:
238:         // Go through provided line and assign values appropriately with switch.
239:         while ((token = strsep(&temp_song, ",")) != NULL) {
240:             switch(song_field_index) {
241:                 case 3: // Album name.
242:                     temp_string = strtok(songs_array[song_index]->album_name, "\\");
243:                     songs_array[song_index]->album_name = copy_string(temp_string, token);
244:                     break;
245:                 case 8: // Artist name.
246:                     temp_string = strtok(songs_array[song_index]->artist, "\\");
247:                     songs_array[song_index]->artist = copy_string(temp_string, token);
248:                     break;
249:                 case 10: // Duration.

```

```
250:         temp_float = atof(token);
251:         temp_float_ptr = &temp_float;
252:         songs_array[song_index]->duration = copy_float(songs_array[song_index]->duration, temp_float_ptr);
253:         break;
254:     case 14: // Hottness.
255:         temp_double = atof(token);
256:         // Check if NAN.
257:         if (temp_double != temp_double) {
258:             temp_double = 0;
259:         }
260:         temp_double_ptr = &temp_double;
261:         songs_array[song_index]->hottnesss = copy_double(songs_array[song_index]->hottnesss, temp_double_ptr);
262:         break;
263:     case 17: // Song name.
264:         temp_string = strtok(songs_array[song_index]->song_name, "\\");
265:         songs_array[song_index]->song_name = copy_string(temp_string, token);
266:         break;
267:     case 18: // Year.
268:         temp_int = atoi(token);
269:         temp_int_ptr = &temp_int;
270:         songs_array[song_index]->year = copy_int(songs_array[song_index]->year, temp_int_ptr);
271:         break;
272:     default: // .
273:         // Do nothing. This is not a field we care about.
274:         break;
275:     }
276:     song_field_index++;
277: }
278: song_index++;
279: }
280: sort_array();
281: }
282:
283:
284: /**
285:  * Converts provided string to lowercase.
286:  * Used to help eliminate errors in user's input.
287:  *
288:  * Return lowercase version of initial string.
289:  */
290: char* to_lower_case(char* input_string) {
291:     int index = 0;
292:     char* return_string = calloc(strlen((input_string) + 1), sizeof(char));
293:     while (input_string[index]) {
294:         return_string[index] = tolower(input_string[index]);
295:         index++;
296:     }
297:     return return_string;
298: }
299:
300:
301: /**
302:  * Increase current buffer size and rebuffer songs array.
303:  * Call to initialize, and then anytime the array runs out of space.
304:  */
305: void buffer_array() {
306:     songs_array_items = songs_array_items * 2;
307:
308:     songs_array = realloc(songs_array, songs_array_items * sizeof(songs_struct* ));
```

```
309:     if (songs_array != NULL) {
310:         // printf("Allocation successful.\n");
311:         // printf("First Album Name: %s\n", songs_array[1]->album_name);
312:     } else {
313:         err_dump("Could not allocate memory for realloc.");
314:     }
315: }
316:
317:
318: /**
319:  * Copies string from destination to source.
320:  */
321: char* copy_string(char* dest_ptr, char* source_ptr) {
322:     dest_ptr = calloc((strlen(source_ptr) + 1), sizeof(char));
323:     if (dest_ptr != NULL) {
324:         memcpy(dest_ptr, source_ptr, ((strlen(source_ptr) + 1) * sizeof(char)));
325:     } else {
326:         err_dump("Could not allocate memory for string calloc.");
327:     }
328:     return dest_ptr;
329: }
330:
331:
332: /**
333:  * Copies float from destination to source.
334:  */
335: int* copy_int(int* dest_ptr, int* source_ptr) {
336:     dest_ptr = calloc(1, sizeof(int));
337:     if (dest_ptr != NULL) {
338:         memcpy(dest_ptr, source_ptr, (sizeof(int*)));
339:     } else {
340:         err_dump("Could not allocate memory for int calloc.");
341:     }
342:     return dest_ptr;
343: }
344:
345:
346: /**
347:  * Copies float from destination to source.
348:  */
349: float* copy_float(float* dest_ptr, float* source_ptr) {
350:     dest_ptr = calloc(1, sizeof(float));
351:     if (dest_ptr != NULL) {
352:         memcpy(dest_ptr, source_ptr, (sizeof(float*)));
353:     } else {
354:         err_dump("Could not allocate memory for float calloc.");
355:     }
356:     return dest_ptr;
357: }
358:
359:
360: /**
361:  * Copies double from destination to source.
362:  */
363: double* copy_double(double* dest_ptr, double* source_ptr) {
364:     dest_ptr = calloc(1, sizeof(double));
365:     if (dest_ptr != NULL) {
366:         memcpy(dest_ptr, source_ptr, (sizeof(double*)));
367:     } else {
368:         err_dump("Could not allocate memory for double calloc.");
369:     }
370:     return dest_ptr;
371: }
```

```

372:
373:
374: /**
375:  * Sorts array by song name.
376:  */
377: void sort_array() {
378:     int sorted_bool = 0;    // Holds if array has been sorted during this call.
379:     int index = 1;         // Current index.
380:     songs_struct* temp_song;
381:
382:     // Loop through all array elements and sort if necessary.
383:     // Starts at 1 due to comparing two elements at once.
384:     while (songs_array[index] != NULL) {
385:         if (strcmp(songs_array[index - 1]->song_name, songs_array[index]->song_name) >
0) {
386:             temp_song = songs_array[index - 1];
387:             songs_array[index - 1] = songs_array[index];
388:             songs_array[index] = temp_song;
389:             sorted_bool = 1;
390:         }
391:         index++;
392:     }
393:
394:     if (sorted_bool == 1) {
395:         sort_array();
396:     }
397: }
398:
399:
400: /**
401:  * Uses binary search to locate song with desired name.
402:  *
403:  * Return either valid song or NULL, depending on if record is found or not.
404:  */
405: songs_struct* search_array_by_song(int first_index, int last_index, char* desired_recor
d) {
406:     songs_struct* return_song = calloc(1, sizeof(songs_struct*)); // Song to return.
407:
408:     // First check if this is the end of search or not.
409:     if (first_index >= last_index) {
410:
411:         // End of search. Handle accordingly.
412:         if ((strcmp(desired_record, to_lower_case(songs_array[first_index]->song_name))
) == 0) {
413:             // Match found. Use given struct.
414:             return_song = songs_array[first_index];
415:         } else {
416:             // End of search and no match found. Use NULL.
417:             return_song = NULL;
418:         }
419:
420:     } else {
421:
422:         // More potential values to search. Handle accordingly.
423:         int mid_index = (first_index + last_index) / 2;
424:         if ((strcmp(desired_record, to_lower_case(songs_array[mid_index]->song_name)))
== 0) {
425:             // Match found. Use given struct.
426:             return_song = songs_array[mid_index];
427:         } else {
428:             if ((strcmp(desired_record, to_lower_case(songs_array[mid_index]->song_name
))) < 0) {
429:                 // Current struct song name is higher letter than desired.

```

```
430:         // Searching first half of provided structs.
431:         return_song = search_array_by_song(first_index, mid_index, desired_reco
rd);
432:     } else {
433:         // Current struct song name is lower letter than desired.
434:         // Searching later half of provided structs.
435:         return_song = search_array_by_song(mid_index + 1, last_index, desired_r
ecord);
436:     }
437: }
438: }
439:
440: return return_song;
441: }
442:
443:
444: /**
445:  * Songs Array is not sorted by album, thus uses basic linear search.
446:  * This is called far less often so that should be acceptable.
447:  *
448:  * Return either valid album or NULL, depending on if record is found or not.
449:  */
450: songs_struct* search_array_by_album(int first_index, int last_index, char* desired_reco
rd) {
451:     // int index = 0;
452:     // while (songs_array[index] != NULL) {
453:     //     if (strcmp(to_lower_case(songs_array[index]->album_name), desired_record) ==
0) {
454:         //         return songs_array[index];
455:         //     }
456:         //     index++;
457:     // }
458:
459:     return NULL;
460: }
461:
462:
463: // User display helper methods.
464:
465: /**
466:  * Prints helper text.
467:  * Called on program start and again if user types "Help".
468:  */
469: void print_help_text() {
470:     printf("To display help text again, type 'Help'.\n");
471:     printf("To print all songs, type 'All'\n");
472:     printf("To exit program, type 'ZZZ'.\n");
473:     printf("Otherwise, type name of song you wish to locate.\n\n");
474: }
475:
476:
477: /**
478:  * Searches through array and finds song based on user input.
479:  *
480:  * Return a valid song or null.
481:  */
482: void find_song(char* user_input_string) {
483:     int counter = -1; // To account for arrays starting at 0.
484:     int index = 0;
485:     while (songs_array[index] != NULL) {
486:         counter++;
487:         index++;
488:     }
```



```
489: // Search for user's input, first by song_name, then by album name.
490: songs_struct* song = search_array_by_song(0, counter, user_input_string);
491: if (song != NULL) {
492:     print_song_info(song);
493: } else { // Not found. Search again by album instead of song_name.
494:     song = search_array_by_album(0, counter, user_input_string);
495:     if (song != NULL) {
496:         print_song_info(song);
497:     } else {
498:         // Could not find song or album.
499:         err_msg("Could not find song or album with name: %s\n", user_input_string);
500:     }
501: }
502: }
503:
504:
505: /**
506:  * Prints all songs.
507:  */
508: void print_all_songs() {
509:     int index = 0;
510:     while (songs_array[index] != NULL) {
511:         print_song_info(songs_array[index]);
512:         index++;
513:     }
514: }
515:
516:
517: /**
518:  * Attempts to find song with provided name. Prints result.
519:  */
520: void print_song_info(songs_struct* song) {
521:     float duration = *song->duration;
522:     int year = *song->year;
523:     double hottnesss = *song->hottnesss;
524:
525:     printf("Song Name: %s\n", song->song_name);
526:     printf("Album Name: %s\n", song->album_name);
527:     printf("Artist Name: %s\n", song->artist);
528:     printf("Duration: %.2f\n", duration);
529:     printf("year: %d\n", year);
530:     printf("Hottnesss: %f\n", hottnesss);
531:     printf("\n");
532: }
```

```
1: gcc -Wall -Wpedantic -std=c99 *.c -g -o 3240Assignment0
2: Main.c: In function â\200\230initialize_dataâ\200\231:
3: Main.c:221:16: warning: cast from pointer to integer of different size [-Wpointer-to-in
t-cast]
4:         temp_int = (int) calloc(1, sizeof(int));
5:                     ^
6: Main.c:234:21: warning: implicit declaration of function â\200\230strdupâ\200\231 [-Wim
plicit-function-declaration]
7:         temp_song = strdup(buffer);
8:                     ^
9: Main.c:234:19: warning: assignment makes pointer from integer without a cast [-Wint-con
version]
10:         temp_song = strdup(buffer);
11:                    ^
12: Main.c:239:25: warning: implicit declaration of function â\200\230strsepâ\200\231 [-Wim
plicit-function-declaration]
13:         while ((token = strsep(&temp_song, ",")) != NULL) {
14:                     ^
15: Main.c:239:23: warning: assignment makes pointer from integer without a cast [-Wint-con
version]
16:         while ((token = strsep(&temp_song, ",")) != NULL) {
17:                     ^
18: Could not find song or album with name: "don't mess with the irs"
19:
20: Could not find song or album with name: "dr. elmo's twisted christmas"
21:
22: Could not find song or album with name: "get on top (album version)"
23:
```

```
1: ./3240Assignment0 < test-input.txt
2: Initializing...
3: Reading csv...
4: To display help text again, type 'Help'.
5: To print all songs, type 'All'
6: To exit program, type 'ZZZ'.
7: Otherwise, type name of song you wish to locate.
8:
9: Song Name: "We're Not Gonna Bow"
10: Album Name: "Ordinary Day"
11: Artist Name: "Jeff And Sheri Easter"
12: Duration: 222.93
13: year: 0
14: Hotttnesss: 0.240821
15:
16: Song Name: "Deep Sea Creature"
17: Album Name: "Call of the Mastodon"
18: Artist Name: "Mastodon"
19: Duration: 280.22
20: year: 2001
21: Hotttnesss: 0.597641
22:
23: Song Name: "Stand By Me"
24: Album Name: "Made In England"
25: Artist Name: "Atomic Rooster"
26: Duration: 203.31
27: year: 1972
28: Hotttnesss: 0.000000
29:
30: Song Name: "Ralph's Rhapsody"
31: Album Name: "The Best Of Ray Lynch"
32: Artist Name: "Ray Lynch"
33: Duration: 283.74
34: year: 1998
35: Hotttnesss: 0.547953
36:
37: Song Name: "Too Many Choices"
38: Album Name: "Personal Business (Explicit)"
39: Artist Name: "Bad Azz"
40: Duration: 285.94
41: year: 0
42: Hotttnesss: 0.405116
43:
44: Song Name: "These Days"
45: Album Name: "Can't Fight Robots"
46: Artist Name: "Take It Back!"
47: Duration: 204.90
48: year: 2008
49: Hotttnesss: 0.447136
50:
51: Song Name: "Armageddon's Raid"
52: Album Name: "Bondage Goat Zombie"
53: Artist Name: "Belphegor"
54: Duration: 308.77
55: year: 2008
56: Hotttnesss: 0.624840
57:
58: Song Name: "Single Ladies (Put A Ring On It)"
59: Album Name: "Single Ladies (Put A Ring On It) - Dance Remixes"
60: Artist Name: "BeyoncÃ©"
61: Duration: 466.05
62: year: 2008
63: Hotttnesss: 0.000000
```

Excellent Input

64:
65: Song Name: "Rudeboy"
66: Album Name: "Aswad vs. The Rhythm Riders"
67: Artist Name: "Aswad"
68: Duration: 258.43
69: year: 0
70: Hottnesss: 0.000000
71:
72: Exiting Program...

```
1: ==19095== Memcheck, a memory error detector
2: ==19095== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
3: ==19095== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
4: ==19095== Command: ./3240Assignment0
5: ==19095==
6: ==19095== Invalid write of size 8 increase amount you malloc
7: ==19095==   at 0x401509: copy_float (Main.c:352) to get rid of this
8: ==19095==   by 0x401178: initialize_data (Main.c:252)
9: ==19095==   by 0x400D68: main (Main.c:148)
10: ==19095== Address 0x5206ed0 is 0 bytes inside a block of size 4 alloc'd
11: ==19095==   at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
12: ==19095==   by 0x4014F2: copy_float (Main.c:350)
13: ==19095==   by 0x401178: initialize_data (Main.c:252)
14: ==19095==   by 0x400D68: main (Main.c:148)
15: ==19095==
16: ==19095== Invalid write of size 8
17: ==19095==   at 0x4014BA: copy_int (Main.c:338)
18: ==19095==   by 0x4012BB: initialize_data (Main.c:270)
19: ==19095==   by 0x400D68: main (Main.c:148)
20: ==19095== Address 0x5206fc0 is 0 bytes inside a block of size 4 alloc'd
21: ==19095==   at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
22: ==19095==   by 0x4014A3: copy_int (Main.c:336)
23: ==19095==   by 0x4012BB: initialize_data (Main.c:270)
24: ==19095==   by 0x400D68: main (Main.c:148)
25: ==19095==
26: ==19095== Conditional jump or move depends on uninitialised value(s)
27: ==19095==   at 0x401667: sort_array (Main.c:384)
28: ==19095==   by 0x401319: initialize_data (Main.c:280) use memset on all
29: ==19095==   by 0x400D68: main (Main.c:148) memory you malloc to get rid of
30: ==19095==
31: ==19095== Conditional jump or move depends on uninitialised value(s) this
32: ==19095==   at 0x401667: sort_array (Main.c:384)
33: ==19095==   by 0x40167C: sort_array (Main.c:395)
34: ==19095==   by 0x401319: initialize_data (Main.c:280)
35: ==19095==   by 0x400D68: main (Main.c:148)
36: ==19095==
37: ==19095== Conditional jump or move depends on uninitialised value(s)
38: ==19095==   at 0x401667: sort_array (Main.c:384)
39: ==19095==   by 0x40167C: sort_array (Main.c:395)
40: ==19095==   by 0x40167C: sort_array (Main.c:395)
41: ==19095==   by 0x401319: initialize_data (Main.c:280)
42: ==19095==   by 0x400D68: main (Main.c:148)
43: ==19095==
44: ==19095== Conditional jump or move depends on uninitialised value(s)
45: ==19095==   at 0x401667: sort_array (Main.c:384)
46: ==19095==   by 0x40167C: sort_array (Main.c:395)
47: ==19095==   by 0x40167C: sort_array (Main.c:395)
48: ==19095==   by 0x40167C: sort_array (Main.c:395)
49: ==19095==   by 0x401319: initialize_data (Main.c:280)
50: ==19095==   by 0x400D68: main (Main.c:148)
51: ==19095==
52: ==19095== Invalid write of size 1
53: ==19095==   at 0x401395: to_lower_case (Main.c:294)
54: ==19095==   by 0x400DFA: main (Main.c:162)
55: ==19095== Address 0x5ab0452 is 0 bytes after a block of size 18 alloc'd
56: ==19095==   at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
57: ==19095==   by 0x401366: to_lower_case (Main.c:292)
58: ==19095==   by 0x400DFA: main (Main.c:162)
59: ==19095==
60: ==19095== Invalid read of size 1
```

```
61: ==19095== at 0x4C30C33: strcat (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
62: ==19095== by 0x400EF3: main (Main.c:186)
63: ==19095== Address 0x5ab0452 is 0 bytes after a block of size 18 alloc'd
64: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
65: ==19095== by 0x401366: to_lower_case (Main.c:292)
66: ==19095== by 0x400DFA: main (Main.c:162)
67: ==19095==
68: ==19095== Conditional jump or move depends on uninitialised value(s)
69: ==19095== at 0x401871: find_song (Main.c:485)
70: ==19095== by 0x400F2D: main (Main.c:188)
71: ==19095==
72: ==19095== Invalid write of size 1
73: ==19095== at 0x401395: to_lower_case (Main.c:294)
74: ==19095== by 0x401749: search_array_by_song (Main.c:424)
75: ==19095== by 0x401885: find_song (Main.c:490)
76: ==19095== by 0x400F2D: main (Main.c:188)
77: ==19095== Address 0x5ab05ae is 0 bytes after a block of size 14 alloc'd
78: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
79: ==19095== by 0x401366: to_lower_case (Main.c:292)
80: ==19095== by 0x401749: search_array_by_song (Main.c:424)
81: ==19095== by 0x401885: find_song (Main.c:490)
82: ==19095== by 0x400F2D: main (Main.c:188)
83: ==19095==
84: ==19095== Invalid write of size 1
85: ==19095== at 0x401395: to_lower_case (Main.c:294)
86: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
87: ==19095== by 0x401885: find_song (Main.c:490)
88: ==19095== by 0x400F2D: main (Main.c:188)
89: ==19095== Address 0x5ab05fe is 0 bytes after a block of size 14 alloc'd
90: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
91: ==19095== by 0x401366: to_lower_case (Main.c:292)
92: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
93: ==19095== by 0x401885: find_song (Main.c:490)
94: ==19095== by 0x400F2D: main (Main.c:188)
95: ==19095==
96: ==19095== Invalid write of size 1
97: ==19095== at 0x401395: to_lower_case (Main.c:294)
98: ==19095== by 0x401749: search_array_by_song (Main.c:424)
99: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
100: ==19095== by 0x401885: find_song (Main.c:490)
101: ==19095== by 0x400F2D: main (Main.c:188)
102: ==19095== Address 0x5ab06b2 is 0 bytes after a block of size 34 alloc'd
103: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
104: ==19095== by 0x401366: to_lower_case (Main.c:292)
105: ==19095== by 0x401749: search_array_by_song (Main.c:424)
106: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
107: ==19095== by 0x401885: find_song (Main.c:490)
108: ==19095== by 0x400F2D: main (Main.c:188)
109: ==19095==
110: ==19095== Invalid write of size 1
111: ==19095== at 0x401395: to_lower_case (Main.c:294)
112: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
113: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
114: ==19095== by 0x401885: find_song (Main.c:490)
115: ==19095== by 0x400F2D: main (Main.c:188)
116: ==19095== Address 0x5ab0722 is 0 bytes after a block of size 34 alloc'd
117: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
```

```
118: ==19095== by 0x401366: to_lower_case (Main.c:292)
119: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
120: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
121: ==19095== by 0x401885: find_song (Main.c:490)
122: ==19095== by 0x400F2D: main (Main.c:188)
123: ==19095==
124: ==19095== Invalid write of size 1
125: ==19095== at 0x401395: to_lower_case (Main.c:294)
126: ==19095== by 0x401749: search_array_by_song (Main.c:424)
127: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
128: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
129: ==19095== by 0x401885: find_song (Main.c:490)
130: ==19095== by 0x400F2D: main (Main.c:188)
131: ==19095== Address 0x5ab07cb is 0 bytes after a block of size 11 alloc'd
132: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
133: ==19095== by 0x401366: to_lower_case (Main.c:292)
134: ==19095== by 0x401749: search_array_by_song (Main.c:424)
135: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
136: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
137: ==19095== by 0x401885: find_song (Main.c:490)
138: ==19095== by 0x400F2D: main (Main.c:188)
139: ==19095==
140: ==19095== Invalid write of size 1
141: ==19095== at 0x401395: to_lower_case (Main.c:294)
142: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
143: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
144: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
145: ==19095== by 0x401885: find_song (Main.c:490)
146: ==19095== by 0x400F2D: main (Main.c:188)
147: ==19095== Address 0x5ab081b is 0 bytes after a block of size 11 alloc'd
148: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
149: ==19095== by 0x401366: to_lower_case (Main.c:292)
150: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
151: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
152: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
153: ==19095== by 0x401885: find_song (Main.c:490)
154: ==19095== by 0x400F2D: main (Main.c:188)
155: ==19095==
156: ==19095== Invalid write of size 1
157: ==19095== at 0x401395: to_lower_case (Main.c:294)
158: ==19095== by 0x401749: search_array_by_song (Main.c:424)
159: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
160: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
161: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
162: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
163: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
164: ==19095== by 0x401885: find_song (Main.c:490)
165: ==19095== by 0x400F2D: main (Main.c:188)
166: ==19095== Address 0x5ab0ac6 is 0 bytes after a block of size 22 alloc'd
167: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
168: ==19095== by 0x401366: to_lower_case (Main.c:292)
169: ==19095== by 0x401749: search_array_by_song (Main.c:424)
170: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
171: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
172: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
173: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
174: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
175: ==19095== by 0x401885: find_song (Main.c:490)
176: ==19095== by 0x400F2D: main (Main.c:188)
177: ==19095==
```

```
178: ==19095== Invalid write of size 1
179: ==19095==   at 0x401395: to_lower_case (Main.c:294)
180: ==19095==   by 0x40179F: search_array_by_song (Main.c:428)
181: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
182: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
183: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
184: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
185: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
186: ==19095==   by 0x401885: find_song (Main.c:490)
187: ==19095==   by 0x400F2D: main (Main.c:188)
188: ==19095== Address 0x5ab0b26 is 0 bytes after a block of size 22 alloc'd
189: ==19095==   at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
190: ==19095==   by 0x401366: to_lower_case (Main.c:292)
191: ==19095==   by 0x40179F: search_array_by_song (Main.c:428)
192: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
193: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
194: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
195: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
196: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
197: ==19095==   by 0x401885: find_song (Main.c:490)
198: ==19095==   by 0x400F2D: main (Main.c:188)
199: ==19095==
200: ==19095== Invalid write of size 1
201: ==19095==   at 0x401395: to_lower_case (Main.c:294)
202: ==19095==   by 0x401749: search_array_by_song (Main.c:424)
203: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
204: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
205: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
206: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
207: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
208: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
209: ==19095==   by 0x401885: find_song (Main.c:490)
210: ==19095==   by 0x400F2D: main (Main.c:188)
211: ==19095== Address 0x5ab0bd7 is 0 bytes after a block of size 23 alloc'd
212: ==19095==   at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
213: ==19095==   by 0x401366: to_lower_case (Main.c:292)
214: ==19095==   by 0x401749: search_array_by_song (Main.c:424)
215: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
216: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
217: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
218: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
219: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
220: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
221: ==19095==   by 0x401885: find_song (Main.c:490)
222: ==19095==   by 0x400F2D: main (Main.c:188)
223: ==19095==
224: ==19095== Invalid write of size 1
225: ==19095==   at 0x401395: to_lower_case (Main.c:294)
226: ==19095==   by 0x40179F: search_array_by_song (Main.c:428)
227: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
228: ==19095==   by 0x4017C8: search_array_by_song (Main.c:431)
229: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
230: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
231: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
232: ==19095==   by 0x4017E4: search_array_by_song (Main.c:435)
233: ==19095==   by 0x401885: find_song (Main.c:490)
234: ==19095==   by 0x400F2D: main (Main.c:188)
235: ==19095== Address 0x5ab0c37 is 0 bytes after a block of size 23 alloc'd
236: ==19095==   at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
237: ==19095==   by 0x401366: to_lower_case (Main.c:292)
```



```
238: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
239: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
240: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
241: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
242: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
243: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
244: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
245: ==19095== by 0x401885: find_song (Main.c:490)
246: ==19095== by 0x400F2D: main (Main.c:188)
247: ==19095==
248: ==19095== Invalid write of size 1
249: ==19095== at 0x401395: to_lower_case (Main.c:294)
250: ==19095== by 0x401749: search_array_by_song (Main.c:424)
251: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
252: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
253: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
254: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
255: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
256: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
257: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
258: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
259: ==19095== by 0x401885: find_song (Main.c:490)
260: ==19095== by 0x400F2D: main (Main.c:188)
261: ==19095== Address 0x5ab0df2 is 0 bytes after a block of size 18 alloc'd
262: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
263: ==19095== by 0x401366: to_lower_case (Main.c:292)
264: ==19095== by 0x401749: search_array_by_song (Main.c:424)
265: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
266: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
267: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
268: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
269: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
270: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
271: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
272: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
273: ==19095== by 0x401885: find_song (Main.c:490)
274: ==19095==
275: ==19095== Invalid write of size 1
276: ==19095== at 0x401395: to_lower_case (Main.c:294)
277: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
278: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
279: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
280: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
281: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
282: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
283: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
284: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
285: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
286: ==19095== by 0x401885: find_song (Main.c:490)
287: ==19095== by 0x400F2D: main (Main.c:188)
288: ==19095== Address 0x5ab0e52 is 0 bytes after a block of size 18 alloc'd
289: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
290: ==19095== by 0x401366: to_lower_case (Main.c:292)
291: ==19095== by 0x40179F: search_array_by_song (Main.c:428)
292: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
293: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
294: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
295: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
296: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
297: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
298: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
```

```
299: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
300: ==19095==      by 0x401885: find_song (Main.c:490)
301: ==19095==
302: ==19095== Invalid read of size 1
303: ==19095==      at 0x4C31FB7: strcmp (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
304: ==19095==      by 0x40175B: search_array_by_song (Main.c:424)
305: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
306: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
307: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
308: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
309: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
310: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
311: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
312: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
313: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
314: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
315: ==19095== Address 0x5ab1244 is 0 bytes after a block of size 20 alloc'd
316: ==19095==      at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
317: ==19095==      by 0x401366: to_lower_case (Main.c:292)
318: ==19095==      by 0x401749: search_array_by_song (Main.c:424)
319: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
320: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
321: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
322: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
323: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
324: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
325: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
326: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
327: ==19095==      by 0x4017E4: search_array_by_song (Main.c:435)
328: ==19095==
329: ==19095== Invalid write of size 1
330: ==19095==      at 0x401395: to_lower_case (Main.c:294)
331: ==19095==      by 0x401749: search_array_by_song (Main.c:424)
332: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
333: ==19095==      by 0x401885: find_song (Main.c:490)
334: ==19095==      by 0x400F2D: main (Main.c:188)
335: ==19095== Address 0x5ab1592 is 0 bytes after a block of size 18 alloc'd
336: ==19095==      at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
337: ==19095==      by 0x401366: to_lower_case (Main.c:292)
338: ==19095==      by 0x401749: search_array_by_song (Main.c:424)
339: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
340: ==19095==      by 0x401885: find_song (Main.c:490)
341: ==19095==      by 0x400F2D: main (Main.c:188)
342: ==19095==
343: ==19095== Invalid write of size 1
344: ==19095==      at 0x401395: to_lower_case (Main.c:294)
345: ==19095==      by 0x40179F: search_array_by_song (Main.c:428)
346: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
347: ==19095==      by 0x401885: find_song (Main.c:490)
348: ==19095==      by 0x400F2D: main (Main.c:188)
349: ==19095== Address 0x5ab15f2 is 0 bytes after a block of size 18 alloc'd
350: ==19095==      at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
351: ==19095==      by 0x401366: to_lower_case (Main.c:292)
352: ==19095==      by 0x40179F: search_array_by_song (Main.c:428)
353: ==19095==      by 0x4017C8: search_array_by_song (Main.c:431)
354: ==19095==      by 0x401885: find_song (Main.c:490)
355: ==19095==      by 0x400F2D: main (Main.c:188)
356: ==19095==
357: ==19095== Invalid read of size 1
```

```
358: ==19095== at 0x4C31FB7: strcmp (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
359: ==19095== by 0x40175B: search_array_by_song (Main.c:424)
360: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
361: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
362: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
363: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
364: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
365: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
366: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
367: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
368: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
369: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
370: ==19095== Address 0x5ab1f72 is 0 bytes after a block of size 18 alloc'd
371: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
372: ==19095== by 0x401366: to_lower_case (Main.c:292)
373: ==19095== by 0x401749: search_array_by_song (Main.c:424)
374: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
375: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
376: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
377: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
378: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
379: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
380: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
381: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
382: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
383: ==19095==
384: ==19095== Invalid read of size 1
385: ==19095== at 0x4C31FB7: strcmp (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
386: ==19095== by 0x40175B: search_array_by_song (Main.c:424)
387: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
388: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
389: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
390: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
391: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
392: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
393: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
394: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
395: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
396: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
397: ==19095== Address 0x5ab2e4c is 0 bytes after a block of size 12 alloc'd
398: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
399: ==19095== by 0x401366: to_lower_case (Main.c:292)
400: ==19095== by 0x401749: search_array_by_song (Main.c:424)
401: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
402: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
403: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
404: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
405: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
406: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
407: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
408: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
409: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
410: ==19095==
411: ==19095== Invalid write of size 1
412: ==19095== at 0x401395: to_lower_case (Main.c:294)
413: ==19095== by 0x4016CF: search_array_by_song (Main.c:412)
414: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
415: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
416: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
```

```
417: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
418: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
419: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
420: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
421: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
422: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
423: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
424: ==19095== Address 0x5ab5a6a is 0 bytes after a block of size 10 alloc'd
425: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
426: ==19095== by 0x401366: to_lower_case (Main.c:292)
427: ==19095== by 0x4016CF: search_array_by_song (Main.c:412)
428: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
429: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
430: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
431: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
432: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
433: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
434: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
435: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
436: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
437: ==19095==
438: Could not find song or album with name: "dr. elmo's twisted christmas"
439:
440: ==19095== Invalid read of size 1
441: ==19095== at 0x4C31FB7: strcmp (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
442: ==19095== by 0x40175B: search_array_by_song (Main.c:424)
443: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
444: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
445: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
446: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
447: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
448: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
449: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
450: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
451: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
452: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
453: ==19095== Address 0x5ab9271 is 0 bytes after a block of size 33 alloc'd
454: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
455: ==19095== by 0x401366: to_lower_case (Main.c:292)
456: ==19095== by 0x401749: search_array_by_song (Main.c:424)
457: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
458: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
459: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
460: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
461: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
462: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
463: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
464: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
465: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
466: ==19095==
467: ==19095== Invalid write of size 1
468: ==19095== at 0x401395: to_lower_case (Main.c:294)
469: ==19095== by 0x4016CF: search_array_by_song (Main.c:412)
470: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
471: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
472: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
473: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
474: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
475: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
476: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
```

```
477: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
478: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
479: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
480: ==19095== Address 0x5aba2a6 is 0 bytes after a block of size 22 alloc'd
481: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
482: ==19095== by 0x401366: to_lower_case (Main.c:292)
483: ==19095== by 0x4016CF: search_array_by_song (Main.c:412)
484: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
485: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
486: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
487: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
488: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
489: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
490: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
491: ==19095== by 0x4017E4: search_array_by_song (Main.c:435)
492: ==19095== by 0x4017C8: search_array_by_song (Main.c:431)
493: ==19095==
494: Could not find song or album with name: "get on top (album version)"
495:
496: ==19095== Invalid read of size 1
497: ==19095== at 0x4C31FB4: strcmp (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
498: ==19095== by 0x400E33: main (Main.c:170)
499: ==19095== Address 0x5abb142 is 0 bytes after a block of size 2 alloc'd
500: ==19095== at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.
so)
501: ==19095== by 0x401366: to_lower_case (Main.c:292)
502: ==19095== by 0x400DFA: main (Main.c:162)
503: ==19095==
504: ==19095==
505: ==19095== HEAP SUMMARY:
506: ==19095== in use at exit: 3,248,356 bytes in 80,508 blocks
507: ==19095== total heap usage: 80,516 allocs, 8 frees, 3,384,644 bytes allocated
508: ==19095==
509: ==19095== LEAK SUMMARY:
510: ==19095== definitely lost: 1,860,821 bytes in 10,491 blocks
511: ==19095== indirectly lost: 0 bytes in 0 blocks
512: ==19095== possibly lost: 100 bytes in 1 blocks
513: ==19095== still reachable: 1,387,435 bytes in 70,016 blocks
514: ==19095== suppressed: 0 bytes in 0 blocks
515: ==19095== Rerun with --leak-check=full to see details of leaked memory
516: ==19095==
517: ==19095== For counts of detected and suppressed errors, rerun with: -v
518: ==19095== Use --track-origins=yes to see where uninitialised values come from
519: ==19095== ERROR SUMMARY: 30332 errors from 30 contexts (suppressed: 0 from 0)
```

Need more frees