LCOV - code coverage report

		Hit	Total	Coverage
Current view:	top level - appsecassignment1 - giftcardreader.c (source / functions)			
Test:	cov2.info	**Lines:**	101	174	58.0 %
Date:	2021-02-21 12:50:31	**Functions:**	5	6	83.3 %

```
      Line data    Source code
    1             : /*
    2             :  * Gift Card Reading Application
    3             :  * Original Author: Shoddycorp's Cut-Rate Contracting
    4             :  * Comments added by: Justin Cappos (JAC) and Brendan Dolan-Gavitt (BDG)
    5             :  * Maintainer:
    6             :  * Date: 8 July 2020
    7             :  */
    8             :
    9             :
   10             : #include "giftcard.h"
   11             :
   12             : #include <stdio.h>
   13             : #include <strings.h>
   14             :
   15             : // interpreter for THX-1138 assembly
   16           4 : void animate(char *msg, unsigned char *program) {
   17             :     unsigned char regs[16];
   18           4 :     char *mptr = msg;
   19           4 :     unsigned char *pc = program;
   20           4 :     int i = 0;
   21           4 :     int zf = 0;
   22             :
   23         202 :     while (1) {
   24             :         unsigned char op, arg1, arg2;
   25         198 :         op = *pc;
   26         198 :         arg1 = *(pc+1);
   27         198 :         arg2 = *(pc+2);
   28         198 :         switch (*pc) {
   29             :             case 0x00:
   30         134 :                 break;
   31             :             case 0x01:
   32             :             //if (arg1 < 0 || arg1 > 15)
   33             :             //{
   34             :                //break;
   35             :           // }
   36           2 :                 regs[arg1] = *mptr;
   37           2 :                 break;
   38             :
   39             :             case 0x02:
   40           0 :                 *mptr = regs[arg1];
   41           0 :                 break;
   42             :             case 0x03:
   43           2 :                 mptr += (char)arg1;
   44           2 :                 break;
```

```
45            :                 case 0x04:
46          0 :                     regs[arg2] = arg1;
47          0 :                     break;
48            :                 case 0x05:
49          0 :                     regs[arg1] ^= regs[arg2];
50          0 :                     zf = !regs[arg1];
51          0 :                     break;
52            :                 case 0x06:
53          0 :                     regs[arg1] += regs[arg2];
54          0 :                     zf = !regs[arg1];
55          0 :                     break;
56            :                 case 0x07:
57          2 :                     puts(msg);
58          2 :                     break;
59            :                 case 0x08:
60          0 :                     goto done;
61            :                 case 0x09:
62          2 :                     pc += arg1;
63            :                    // pc += (char)arg1;
64          2 :                     break;
65            :
66            :                 case 0x10:
67          0 :                     if (zf) pc += (char)arg1;
68          0 :                     break;
69            :             }
70        198 :             pc+=3;
71        198 :             if (pc > program+256) break;
72            :         }
73            : done:
74          4 :         return;
75            : }
76            :
77          6 : int get_gift_card_value(struct this_gift_card *thisone) {
78            :             struct gift_card_data *gcd_ptr;
79            :             struct gift_card_record_data *gcrd_ptr;
80            :             struct gift_card_amount_change *gcac_ptr;
81          6 :             int ret_count = 0;
82            :
83          6 :             gcd_ptr = thisone->gift_card_data;
84         12 :             for(int i=0;i<gcd_ptr->number_of_gift_card_records; i++) {
85          6 :                     gcrd_ptr = (struct gift_card_record_data *) gcd_ptr->gift_card_record_data[i];
86          6 :                     if (gcrd_ptr->type_of_record == 1) {
87          0 :                             gcac_ptr = gcrd_ptr->actual_record;
88          0 :                             ret_count += gcac_ptr->amount_added;
89          0 :                     }
90          6 :             }
91          6 :             return ret_count;
92            : }
93            :
94            :
95          6 : void print_gift_card_info(struct this_gift_card *thisone) {
96            :             struct gift_card_data *gcd_ptr;
97            :             struct gift_card_record_data *gcrd_ptr;
98            :             struct gift_card_amount_change *gcac_ptr;
99            :         struct gift_card_program *gcp_ptr;
```

```
100              :
101            6 :         gcd_ptr = thisone->gift_card_data;
102            6 :         printf("   Merchant ID: %32.32s\n",gcd_ptr->merchant_id);
103            6 :         printf("   Customer ID: %32.32s\n",gcd_ptr->customer_id);
104            6 :         printf("   Num records: %d\n",gcd_ptr->number_of_gift_card_records);
105           12 :         for(int i=0;i<gcd_ptr->number_of_gift_card_records; i++) {
106            6 :                 gcrd_ptr = (struct gift_card_record_data *) gcd_ptr->gift_card_record_data[i];
107            6 :                 if (gcrd_ptr->type_of_record == 1) {
108            0 :                         printf("      record_type: amount_change\n");
109            0 :                         gcac_ptr = gcrd_ptr->actual_record;
110            0 :                         printf("      amount_added: %d\n",gcac_ptr->amount_added);
111            0 :                         if (gcac_ptr->amount_added>0) {
112            0 :                                 printf("      signature: %32.32s\n",gcac_ptr->actual_signature);
113            0 :                         }
114            0 :                 }
115            6 :                 else if (gcrd_ptr->type_of_record == 2) {
116            2 :                         printf("      record_type: message\n");
117            2 :                         printf("      message: %s\n",(char *)gcrd_ptr->actual_record);
118            2 :                 }
119            4 :                 else if (gcrd_ptr->type_of_record == 3) {
120            4 :             gcp_ptr = gcrd_ptr->actual_record;
121            4 :                         printf("      record_type: animated message\n");
122            4 :             printf("      message: %s\n", gcp_ptr->message);
123            4 :             printf("  [running embedded program]  \n");
124            4 :             animate(gcp_ptr->message, gcp_ptr->program);
125            4 :                 }
126            6 :         }
127            6 :         printf("  Total value: %d\n\n",get_gift_card_value(thisone));
128            6 : }
129              :
130              : // Added to support web functionalities
131              :
132              :
133              :
134            0 : void gift_card_json(struct this_gift_card *thisone) {
135              :     struct gift_card_data *gcd_ptr;
136              :     struct gift_card_record_data *gcrd_ptr;
137              :     struct gift_card_amount_change *gcac_ptr;
138            0 :     gcd_ptr = thisone->gift_card_data;
139            0 :     printf("{\n");
140            0 :     printf("  \"merchant_id\": \"%32.32s\",\n", gcd_ptr->merchant_id);
141            0 :     printf("  \"customer_id\": \"%32.32s\",\n", gcd_ptr->customer_id);
142            0 :     printf("  \"total_value\": %d,\n", get_gift_card_value(thisone));
143            0 :     printf("  \"records\": [\n");
144            0 :         for(int i=0;i<gcd_ptr->number_of_gift_card_records; i++) {
145            0 :         gcrd_ptr = (struct gift_card_record_data *) gcd_ptr->gift_card_record_data[i];
146            0 :         printf("    {\n");
147            0 :         if (gcrd_ptr->type_of_record == 1) {
148            0 :             printf("      \"record_type\": \"amount_change\",\n");
149            0 :             gcac_ptr = gcrd_ptr->actual_record;
150            0 :             printf("      \"amount_added\": %d,\n",gcac_ptr->amount_added);
151            0 :             if (gcac_ptr->amount_added>0) {
152            0 :                 printf("      \"signature\": \"%32.32s\"\n",gcac_ptr->actual_signature);
153            0 :             }
154            0 :         }
```

```
155          0 :         else if (gcrd_ptr->type_of_record == 2) {
156          0 :                         printf("      \"record_type\": \"message\",\n");
157          0 :                         printf("      \"message\": \"%s\"\n",(char *)gcrd_ptr->actual_record);
158          0 :         }
159          0 :         else if (gcrd_ptr->type_of_record == 3) {
160          0 :             struct gift_card_program *gcp = gcrd_ptr->actual_record;
161          0 :                         printf("      \"record_type\": \"animated message\",\n");
162          0 :                         printf("      \"message\": \"%s\",\n",gcp->message);
163            :             // programs are binary so we will hex for the json
164          0 :             char *hexchars = "01234567890abcdef";
165            :             char program_hex[512+1];
166          0 :             program_hex[512] = '\0';
167            :             int i;
168          0 :             for(i = 0; i < 256; i++) {
169          0 :                 program_hex[i*2] = hexchars[((gcp->program[i] & 0xf0) >> 4)];
170          0 :                 program_hex[i*2+1] = hexchars[(gcp->program[i] & 0x0f)];
171          0 :             }
172          0 :                         printf("      \"program\": \"%s\"\n",program_hex);
173          0 :         }
174          0 :         if (i < gcd_ptr->number_of_gift_card_records-1)
175          0 :             printf("    },\n");
176            :         else
177          0 :             printf("    }\n");
178          0 :     }
179          0 :     printf("  ]\n");
180          0 :     printf("}\n");
181          0 : }
182            :
183            :
184            :
185            :
186            :
187            : /* JAC: input_fd is misleading... It's a FILE type, not a fd */
188          8 : struct this_gift_card *gift_card_reader(FILE *input_fd) {
189            :
190          8 :         struct this_gift_card *ret_val = malloc(sizeof(struct this_gift_card));
191            :
192            :     void *optr;
193            :         void *ptr;
194            :
195            :         // Loop to do the whole file
196         14 :         while (!feof(input_fd)) {
197            :
198            :                 struct gift_card_data *gcd_ptr;
199            :                 /* JAC: Why aren't return types checked? */
200          8 :                 fread(&ret_val->num_bytes, 4,1, input_fd);
201            :
202          8 :         if (ret_val->num_bytes < 0) {
203            :
204          2 :             printf (" Negative value\n");
205          2 :             exit(0);
206            :
207            :         }
208            :
209            :                 // Make something the size of the rest and read it in
```

```
  210          6 :                 ptr = malloc(ret_val->num_bytes);
  211          6 :                 fread(ptr, ret_val->num_bytes, 1, input_fd);
  212            :
  213          6 :         optr = ptr-4;
  214            :
  215          6 :                 gcd_ptr = ret_val->gift_card_data = malloc(sizeof(struct gift_card_data));
  216          6 :                 gcd_ptr->merchant_id = ptr;
  217          6 :                 ptr += 32;
  218            : //              printf("VD: %d\n",(int)ptr - (int) gcd_ptr->merchant_id);
  219          6 :                 gcd_ptr->customer_id = ptr;
  220          6 :                 ptr += 32;
  221            :                 /* JAC: Something seems off here... */
  222          6 :                 gcd_ptr->number_of_gift_card_records = *((char *)ptr);
  223          6 :                 ptr += 4;
  224            :
  225          6 :                 gcd_ptr->gift_card_record_data = (void *)malloc(gcd_ptr->number_of_gift_card_records*sizeof(void*));
  226            :
  227            :                 // Now ptr points at the gift card recrod data
  228         18 :                 for (int i=0; i<=gcd_ptr->number_of_gift_card_records; i++){
  229            :                         //printf("i: %d\n",i);
  230            :                         struct gift_card_record_data *gcrd_ptr;
  231         12 :                         gcrd_ptr = gcd_ptr->gift_card_record_data[i] = malloc(sizeof(struct gift_card_record_data));
  232            :                         struct gift_card_amount_change *gcac_ptr;
  233         12 :                         gcac_ptr = gcrd_ptr->actual_record = malloc(sizeof(struct gift_card_record_data));
  234            :             struct gift_card_program *gcp_ptr;
  235         12 :                         gcp_ptr = malloc(sizeof(struct gift_card_program));
  236            :
  237         12 :                         gcrd_ptr->record_size_in_bytes = *((char *)ptr);
  238            :             //printf("rec at %x, %d bytes\n", ptr - optr, gcrd_ptr->record_size_in_bytes);
  239         12 :                         ptr += 4;
  240            :                         //printf("record_data: %d\n",gcrd_ptr->record_size_in_bytes);
  241         12 :                         gcrd_ptr->type_of_record = *((char *)ptr);
  242         12 :                         ptr += 4;
  243            :             //printf("type of rec: %d\n", gcrd_ptr->type_of_record);
  244            :
  245            :                         // amount change
  246         12 :                         if (gcrd_ptr->type_of_record == 1) {
  247          0 :                                 gcac_ptr->amount_added = *((int*) ptr);
  248          0 :                                 ptr += 4;
  249            :
  250            :                                 // don't need a sig if negative
  251            :                                 /* JAC: something seems off here */
  252          0 :                                 if (gcac_ptr < 0) break;
  253            :
  254          0 :                                 gcac_ptr->actual_signature = ptr;
  255          0 :                                 ptr+=32;
  256          0 :                         }
  257            :                         // message
  258         12 :                         if (gcrd_ptr->type_of_record == 2) {
  259          2 :                                 gcrd_ptr->actual_record = ptr;
  260            :                                 // advance by the string size + 1 for nul
  261            :                 // BDG: does not seem right
  262          2 :                                 ptr=ptr+strlen((char *)gcrd_ptr->actual_record)+1;
  263          2 :                         }
  264            :             // BDG: never seen one of these in the wild
```

```
265            :             // text animatino (BETA)
266           12 :             if (gcrd_ptr->type_of_record == 3) {
267            4 :                 gcp_ptr->message = malloc(32);
268            4 :                 gcp_ptr->program = malloc(256);
269            4 :                 memcpy(gcp_ptr->message, ptr, 32);
270            4 :                 ptr+=32;
271            4 :                 memcpy(gcp_ptr->program, ptr, 256);
272            4 :                 ptr+=256;
273            4 :                 gcrd_ptr->actual_record = gcp_ptr;
274            4 :             }
275           12 :                 }
276            :         }
277            6 :         return ret_val;
278            : }
279            :
280            : // BDG: why not a local variable here?
281            : struct this_gift_card *thisone;
282            :
283            8 : int main(int argc, char **argv) {
284            :     // BDG: no argument checking?
285            8 :         FILE *input_fd = fopen(argv[2],"r");
286            8 :     if (input_fd == NULL)
287            :     {
288            0 :         printf("sorry, you must pass a gift card file.\n");
289            0 :         return 0;
290            :
291            :     }
292            8 :         thisone = gift_card_reader(input_fd);
293            8 :         if (argv[1][0] == '1') print_gift_card_info(thisone);
294            0 :     else if (argv[1][0] == '2') gift_card_json(thisone);
295            :
296            6 :         return 0;
297            6 : }
```

Generated by: LCOV version 1.15