```
 1  #define _CRT_SECURE_NO_WARNINGS
 2  #include <stdio.h>
 3  #include <stdlib.h>
 4  #include <math.h>
 5  #include <string.h>
 6  #include <ctype.h>
 7  #include <time.h>
 8  #include <windows.h>
 9  #define MAX_LEN 150
10  #define TRUE 1
11  #define FALSE 0
12
13  //List of functions sorted by order of usage
14  void displayTime();
                //Displays time on the upperleft corner of the screen
15
16  int main();
                //Starting point of the program
17  int displayStart();
                //Prints the starting banner and acts as the starting screen
18  void print_image(FILE *fptr);
                //Prints most ASCII arts
19  int displayMenu();
                //Displays the movie selection screen
20
21  int displaySeat(int movieNum);
                //Shows available seats and take user's input
22  int seatPreview(int movieNum);
                //Previews seats
23  int noDupes(char arr[], int size, char input, int count);
                //Makes sure the user doesn't enter a seat they've already
      picked in the same session
24
25  int displaySummary(int movieNum, int NumofSeat, char SeatNum[], char
      seatHolder[]);      //Displays a confirmation message (how many seats you
      picked, which seats you picked)
26  void printArray(char arr[], int size);
                //Prints an array
27
28  int displayTicket(int movieNum, int NumofSeat, char SeatNum[]);
                //Prints out your ticket with printf
29  void print_imageTicket(FILE *fptr);
                //Prints ASCII art (which are aligned differently) present in
      the ticket printing
30
31  int displayFinal();
                //Displays a "thank you" message and taking you back to
      starting screen
32
33  int displayClose();
                //Displays a confirmation message asking whether the user
      really wants to terminate the program.
34
```

```
35 int displayHistory();
                //Displays history of purchase
36 void writeData(int movieNum, int NumofSeat, char SeatNum[]);
                //Update the file storing the seat data
37 void writeTime();
                //Writes down the time of purchase into history
38
39 //Notes:
40 //usin stands for "User Input"
41
42 void displayTime() { //Shows time on the upperleft corner (based on the
     system's calendar)
43     struct tm *local, *gm;
44     time_t t;
45     t = time(NULL);
46     local = localtime(&t);
47
48     char weekDays[][MAX_LEN] =
         { "Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturda
         y" }; //local->tm_wday returns an int representing what day it is in
         this week, with 0 starting as Sunday
49     char months[][MAX_LEN] =
         { "January","February","March","April","May","June","July","August","S
         eptember","October","November","December" }; //local->tm_mon returns
         an int representing what month it is in this year, with 0 starting as
         January
50
51     printf("%02d:%02d\n", local->tm_hour, local->tm_min); //Hour:Minute
52     printf("%s,%02d %s %02d", weekDays[local->tm_wday], local->tm_mday,
         months[local->tm_mon], local->tm_year + 1900); //Weekday, Date Month
         Year
53     printf("\n\n");
54 }
55
56 int main()
57 {
58     char usin; //usin stands for "User Input"
59     displayStart();
60     printf("\n'H' to access history of purchase // 'X' to terminate
         program");
61     printf("\nPress any key to continue: ");
62     usin = toupper(getchar());
63     while (getchar() != '\n'); //Clearing Buffer
64
65     if (usin == 'X') { displayClose(); }
66     else if (usin == 'H') { displayHistory(); }
67     else { displayMenu(); }
68 }
69
70 int displayStart() {
71     system("cls");
72     displayTime();
73     printf("\t\t\t");
```

```
74
75
76      //Printing ================================
77      for (int i = 0; i < 80; i++) {
78          printf("=");
79          Sleep(10);
80      }
81      //Printing ================================
82
83
84      //Printing "Mayor Cineplex" banner
85      char *filename = "Headertext.txt";
86      FILE *fptr = NULL;
87      if ((fptr = fopen(filename, "r")) == NULL)
88      {
89          printf("Unable to open %s", filename); return 1;
90      }
91      print_image(fptr);
92      fclose(fptr);
93      //Printing "Mayor Cineplex" banner
94
95
96      //Printing ================================
97      printf("\n\t\t\t");
98      for (int i = 0; i < 80; i++) {
99          printf("="); Sleep(10);
100     }
101     //Printing ================================
102 }
103
104 void print_image(FILE *fptr)
105 {
106     char read_string[MAX_LEN];
107     while (fgets(read_string, sizeof(read_string), fptr) != NULL)
108     {
109         printf("\t\t%s", read_string);
110         Sleep(100);
111     }
112 }
113
114 int displayMenu() {
115     int usin; //Accepts the movie number
116     system("cls");
117     displayTime();
118
119     FILE *fptr1 = NULL;
120     FILE *fptr2 = NULL;
121     FILE *fptr3 = NULL;
122     char *filename1 = "Art1.txt";
123     char *filename2 = "Art2.txt";
124     char *filename3 = "Art3.txt";
125
126
```

```
127     if ((fptr1 = fopen(filename1, "r")) == NULL) {  //The file pointer
          points to the seat file of whichever movie you picked
128         printf("Unable to open %s", filename1);
129         return 1;
130     }
131     if ((fptr2 = fopen(filename2, "r")) == NULL) {
132         printf("Unable to open %s", filename2);
133         return 1;
134     }
135     if ((fptr3 = fopen(filename3, "r")) == NULL) {
136         printf("Unable to open %s", filename3);
137         return 1;
138     }
139
140     printf("\n\t"); //Printing ===========
141     for (int i = 0; i < 60; i++) {
142         printf("=");
143         Sleep(10);
144     }
145     printf("\n");
146     print_image(fptr1); //Printing movie posters
147     printf("\n\n\t");
148     for (int i = 0; i < 60; i++) {
149         printf("=");
150         Sleep(10);
151     }
152     printf("\n");
153     print_image(fptr2); //Printing movie posters
154     printf("\n\n\t");
155     for (int i = 0; i < 60; i++) {
156         printf("=");
157         Sleep(10);
158     }
159
160     printf("\n\t");
161     printf("\n");
162     print_image(fptr3); //Printing movie posters
163     printf("\n\n\t");
164
165
166     for (int i = 0; i < 60; i++) {//Printing ===========
167         printf("=");
168         Sleep(10);
169     }
170
171     fclose(fptr1);
172     fclose(fptr2);
173     fclose(fptr3);
174
175     printf("\nPick your movie!\n");
176     printf("<1> <2> <3>\n");
177
178     //ACCEPTING USER INPUT
```

```
179     printf("Note: input '0' to go back\n");
180     while (1) {
181
182         printf("Your input: ");
183         scanf("%d", &usin);
184         if (usin == 0) { //Inputting a '0' will take you back to the start
185             while (getchar() != '\n'); //Clearing buffer
186             system("cls");
187             printf("Taking you back to start.");
188             for (int i = 0; i < 3; i++) {
189                 printf(".");
190                 Sleep(1000);
191             }
192             return main();
193         }
194         if (usin == 1 || usin == 2 || usin == 3) { break; } //Only accepts 1
               or 2 or 3
195         printf("Please pick one of the three movies.\n");
196         while (getchar() != '\n');
197     }
198     //ACCEPTING USER INPUT
199
200
201     printf("You picked movie number %d", usin);
202
203
204     //LOADING BAR %|=======|%
205     printf("\nLoading...\n");
206     for (int loop = 0; loop < 2; loop++) {
207         printf("%%||");
208         for (int i = 0; i < 30; i++) {
209             printf("="); Sleep(25);
210         }
211         printf("||%%");
212         printf("\n");
213     }
214     //LOADING BAR %|=======|%
215
216     system("pause");
217     printf("\n");
218     displaySeat(usin); //Passing the movie number into the DisplaySeat()
        function
219 }
220
221 int displaySeat(int movieNum) {
222     system("cls");
223     displayTime();
224     int usinNum; //Accepts HOW MANY seats the user wants to book
225     char usinSeat[MAX_LEN]; //Accepts WHICH seats the user wants to book
226     char temp;
227     seatPreview(movieNum); //A preview of seats according to what movie the
        user picked
228
```

```
229     //Accept user input
230     printf("Note: input '0' to go back");
231     while (1) {
232         printf("\nHow many seats (up to 5 seats): ");
233         scanf("%d", &usinNum);
234         if (usinNum > 0 && usinNum <= 5) { //The user can pick up to 5 seats
235             break;
236         }
237         else if (usinNum == 0) {
238             system("cls");
239             printf("Taking you back to the movie selection screen."); //
                  Inputting a '0' will take you back to the movie selection
                  screen
240             for (int i = 0; i < 3; i++) {
241                 printf(".");
242                 Sleep(1000);
243             }
244             displayMenu();
245         }
246         getchar();
247         printf("Only integers of value 1-5 are accepted.");
248     }
249     while (getchar() != '\n');
250
251
252     //// PULLS IN THE SEAT DATA
253     char *filename1 = "Seat1.txt";
254     char *filename2 = "Seat2.txt";
255     char *filename3 = "Seat3.txt";
256     char seatHolder[MAX_LEN]; //Accepts the seat data from text file
257     FILE *fptr = NULL;
258
259     if (movieNum == 1) { fptr = fopen(filename1, "r"); } //Points to the
          seat file of whichever movie you picked
260     else if (movieNum == 2) { fptr = fopen(filename2, "r"); }
261     else if (movieNum == 3) { fptr = fopen(filename3, "r"); }
262
263     if (fptr == NULL)
264     {
265         printf("Unable to open file"); return 1; //In case of not being able
              to open the file
266     }
267     else {
268         fscanf(fptr, "%s", seatHolder); //Putting seat data into this
              variable
269         fclose(fptr);
270     }
271     //// PULLS IN THE SEAT DATA
272
273
274     //Accept user input
275     printf("\n");
276     printf("Note - X represents occupied seats\n"); //Occupied seats are
```

```
          represented with 'X'
277     for (int i = 0; i < usinNum;) {
278         printf("Input seat number %d : ", i + 1);
279         scanf("%c", &usinSeat[i]);
280         usinSeat[i] = toupper(usinSeat[i]);
281         temp = usinSeat[i];
282         while (getchar() != '\n');
283 
284         if (!isalpha(temp)) { //Seats are represented as alphabets, so non-
              alphabets are ignored
285             printf("Please input the seat number with alphabets.\n");
286             continue;
287         }
288         else if (temp == 'X') { //X = occupied seat
289             printf("Invalid Input (X represents an occupied seat).\n");
290         }
291         else if (noDupes(usinSeat, usinNum, temp, i) == TRUE) { //No
              duplicate inputs
292             continue;
293         }
294         else if (strchr(seatHolder, temp) != NULL) { //Input matches seat
              (seat available)
295             printf("Seat input: %c\n", usinSeat[i]);
296             i++;
297             for (int i = 0; i < strlen(seatHolder); i++) {
298                 if (seatHolder[i] == temp) {
299                     seatHolder[i] = 'X'; //Replacing with X (occupied seat)
300                     printf("Input successful!\n\n");
301                 }
302             }
303         }
304         else {//Input does not match seat (seat taken)
305             printf("Seat is either taken or non-existent\n");
306             continue;
307         }
308     }
309     usinSeat[usinNum] = '\0'; //Attaching this puts an EOF to the string
310     system("pause");
311     return displaySummary(movieNum, usinNum, usinSeat, seatHolder);
312     //                  Movie number, How many seats, Which seats, Seat data
313 }
314 
315 int seatPreview(int movieNum) { //Preview seats
316     char tempChar; //A placeholder for reading single characters from the
          seat file
317 
318     char *filename1 = "Seat1.txt"; //Seat files store each seat's
          availability
319     char *filename2 = "Seat2.txt";
320     char *filename3 = "Seat3.txt";
321 
322     FILE *fptr = NULL;
323     if (movieNum == 1) { fptr = fopen(filename1, "r"); } //The file pointer
```

```
          points to the seat file of whichever movie you picked
324       else if (movieNum == 2) { fptr = fopen(filename2, "r"); }
325       else if (movieNum == 3) { fptr = fopen(filename3, "r"); }
326
327       if (fptr == NULL)
328       {
329           printf("Unable to open seat preview"); return 1;//In case of not
              being able to open the file
330       }
331       else {
332           int countofThree = 0;
333           int countRow = 0;
334           printf("\t============================================== Screen
              ==============================================\n\n");
335           for (int i = 0; i < 24; i++) { //An indentation(\t) every 3 seats
336               if (countofThree == 3) {
337                   countofThree = 0;
338                   printf("\t");
339               }
340               if (countRow == 6) { //New row every 6 seats with ---------
                  being inbetween each row
341                   countRow = 0;
342                   printf("\n\n
                      \t------------------------------------------------------------
                      ----------------------------------------\n\n");
343               }
344
345               tempChar = fgetc(fptr);
346               printf("\t");
347               Sleep(250); //Putting a 0.25 delay between each printf's makes
                  it more stylish
348               printf("  |   %c   |  ", tempChar);
349               countofThree += 1;
350               countRow += 1;
351           }
352           printf("\n\n
              \t------------------------------------------------------------------
              ----------------------------------\n\n");
353           fclose(fptr);
354       }
355   }
356
357   int noDupes(char arr[], int size, char input, int count) { //TRUE = there is
         a dupe, FALSE = no dupe
358       switch (size) { //Hard coding coming up. This prevents the user from
            entering the same seat input in the same session
359       case 1:
360           return FALSE;
361       case 2:
362           if (count == 0) { return FALSE; }
363           else if (count == 1) {
364               if (input == arr[count - 1]) {
365                   printf("That seat has already been picked.\n");
```

```
366                 return TRUE;
367             }
368         }
369     case 3:
370         if (count == 0) { return FALSE; }
371         else if (count == 1) {
372             if (input == arr[count - 1]) {
373                 printf("That seat has already been picked.\n");
374                 return TRUE;
375             }
376         }
377         else if (count == 2) {
378             if (input == arr[count - 1] || input == arr[count - 2]) {
379                 printf("That seat has already been picked.\n");
380                 return TRUE;
381             }
382         }
383     case 4:
384         if (count == 0) { return FALSE; }
385         else if (count == 1) {
386             if (input == arr[count - 1]) {
387                 printf("That seat has already been picked.\n");
388                 return TRUE;
389             }
390         }
391         else if (count == 2) {
392             if (input == arr[count - 1] || input == arr[count - 2]) {
393                 printf("That seat has already been picked.\n");
394                 return TRUE;
395             }
396         }
397         else if (count == 3) {
398             if (input == arr[count - 1] || input == arr[count - 2] || input
                  == arr[count - 3]) {
399                 printf("That seat has already been picked.\n");
400                 return TRUE;
401             }
402         }
403     case 5:
404         if (count == 0) { return FALSE; }
405         else if (count == 1) {
406             if (input == arr[count - 1]) {
407                 printf("That seat has already been picked.\n");
408                 return TRUE;
409             }
410         }
411         else if (count == 2) {
412             if (input == arr[count - 1] || input == arr[count - 2]) {
413                 printf("That seat has already been picked.\n");
414                 return TRUE;
415             }
416         }
417         else if (count == 3) {
```

```
418             if (input == arr[count - 1] || input == arr[count - 2] || input
                  == arr[count - 3]) {
419                 printf("That seat has already been picked.\n");
420                 return TRUE;
421             }
422         }
423         else if (count == 4) {
424             if (input == arr[count - 1] || input == arr[count - 2] || input
                  == arr[count - 3] || input == arr[count - 4]) {
425                 printf("That seat has already been picked.\n");
426                 return TRUE;
427             }
428         }
429     }
430
431 }
432
433 int displaySummary(int movieNum, int NumofSeat, char SeatNum[], char
      seatHolder[]) {
434     system("cls");
435     displayTime();
436     char usin;
437     char *filename1 = "Seat1.txt";
438     char *filename2 = "Seat2.txt";
439     char *filename3 = "Seat3.txt";
440     FILE *fptr = NULL;
441
442     if (movieNum == 1) { fptr = fopen(filename1, "w"); } //Points to the
          seat file of whichever movie you pick
443     else if (movieNum == 2) { fptr = fopen(filename2, "w"); }
444     else if (movieNum == 3) { fptr = fopen(filename3, "w"); }
445
446     Sleep(250);
447     printf("You picked %d seat(s) (Movie <%d>)!\n", NumofSeat, movieNum);
448     Sleep(250);
449     printf("Here are your seats: ");
450     printArray(SeatNum, NumofSeat); //Print out which seats you picked
451
452     do {
453         printf("\nConfirm? (Y/N): ");
454         usin = toupper(getchar());
455         while (getchar() != '\n');
456     } while ((usin != 'Y') && (usin != 'N'));
457
458     if (usin == 'Y') { //'Y' represents "Yes, I want to confirm my seat(s)."
459
460         if (fptr == NULL)
461         {
462             printf("Unable to open re-write seat"); return 1; //In case of
                  not being able to open the file
463         }
464         else {
465             fprintf(fptr, "%s", seatHolder); //Updates the seat data file
```

```
                with selected seats
466             fclose(fptr);
467             printf("%%||");
468             for (int i = 0; i < 30; i++) {
469                 printf("="); Sleep(70);
470             }
471             printf("%%||");
472             printf("\nSeat Updated!\n"); //Signifies that the process is
                successful
473         }
474
475     }
476     else if (usin == 'N') { //'N' represents "No, take me back to the seat
        selection screen."
477         return displaySeat(movieNum);
478     }
479     system("pause");
480     return displayTicket(movieNum, NumofSeat, SeatNum); //Now we print out
        our ticket
481
482 }
483
484 void printArray(char arr[], int size) {
485     for (int i = 0; i < size; i++) {
486         printf("%c ", arr[i]);
487     }
488 }
489
490 int displayTicket(int movieNum, int NumofSeat, char SeatNum[]) {
491     system("cls");
492     displayTime();
493     printf("Printing ticket...\n");
494
495     char *filename1 = "Art1.txt";
496     char *filename2 = "Art2.txt";
497     char *filename3 = "Art3.txt";
498     FILE *fptr = NULL;
499
500     if (movieNum == 1) { fptr = fopen(filename1, "r"); } //Points to the
        seat file of whichever movie you pick
501     else if (movieNum == 2) { fptr = fopen(filename2, "r"); }
502     else if (movieNum == 3) { fptr = fopen(filename3, "r"); }
503
504
505     printf("\t\t"); //Printing ======== as top of the ticket, making it look
           more refined
506     for (int i = 0; i < 45; i++) {
507         printf("="); Sleep(25);
508     }
509     printf("\n");
510     printf("\t\t   | Movie Number << %d >> | Airing Now!\n", movieNum); //
        Movie number (1-3) that the user picked shows up here
511
```

```
512     print_imageTicket(fptr);
513     fclose(fptr);
514     printf("\n");
515
516     printf("\n\t\t\t%6d Seat(s): ", NumofSeat); //This shows how many seats
          the user picked
517     printArray(SeatNum, NumofSeat);
518     printf("\n");
519     printf("\t\t");
520     for (int i = 0; i < 45; i++) {
521         printf("="); Sleep(25); //Printing ========
522     }
523     printf("\n\n");
524
525     Sleep(1000);
526     printf("Writing data to history..\n"); //Signifies the user that their
          purchase history is being written into the history file
527     printf("%%||");
528     for (int i = 0; i < 30; i++) {
529         printf("="); Sleep(70);
530     }
531     printf("||%%");
532
533     writeTime(); //Write time of purchase and other info into history file
534     writeData(movieNum, NumofSeat, SeatNum);
535
536     printf("\n");
537
538     system("pause");
539     displayFinal();
540 }
541
542 void print_imageTicket(FILE *fptr) //This function is created solely because
       the ASCII on ticket has to be aligned differently from the other ASCII
      arts
543 {
544     char read_string[MAX_LEN];
545     while (fgets(read_string, sizeof(read_string), fptr) != NULL)
546     {
547         printf("\t\t %s", read_string);
548         Sleep(250);
549     }
550 }
551
552 int displayFinal() {
553     system("cls");
554     displayTime();
555
556     //Printing "thank you" banner
557     char *filename = "Thankyou.txt";
558     FILE *fptr = NULL;
559     if ((fptr = fopen(filename, "r")) == NULL)
560     {
```

```
561         printf("Unable to open %s", filename); return 1;
562     }
563     print_image(fptr);
564     printf("\n");
565     fclose(fptr);
566     //Printing "thank you" banner
567 
568     system("pause");
569     return main();
570 }
571 
572 int displayClose() {
573     char usin;
574     FILE *fptr;
575     FILE *fptr_cat;
576     char *filename = "Close.txt";
577     char *filename_cat = "Cat.txt";
578 
579     system("cls");
580     displayTime();
581 
582     if ((fptr_cat = fopen(filename_cat, "r")) == NULL)
583     {
584         printf("error opening %s\n", filename_cat); //In case of the file
              failing to open
585         return 1;
586     }
587 
588     printf("Terminating Console....\n");
589     print_image(fptr_cat); //Prints an image of a cat
590     Sleep(1000);
591     fclose(fptr_cat);
592 
593 
594     if ((fptr = fopen(filename, "r")) == NULL)
595     {
596         printf("error opening %s\n", filename); //In case of the file
              failing to open
597         return 1;
598     }
599 
600 
601     printf("\n");
602     do {
603         printf("You are leaving us? (Y/N): ");
604         usin = toupper(getchar());
605         while (getchar() != '\n');
606     } while ((usin != 'Y') && (usin != 'N'));
607 
608     if (usin == 'Y') {
609         system("cls");
610         printf("\n\n");
611         print_image(fptr); //Prints an image of a plane with the word
```

```
            "Goodbye!"
612         Sleep(1500);
613         fclose(fptr);
614         return 0;
615     }
616     else if (usin == 'N') { return main(); }
617 }
618
619 int displayHistory() {
620     //LOADING SCREEN
621     system("cls");
622     printf("Displaying History of Purchase...\n");
623     printf("%%||");
624     for (int i = 0; i < 30; i++) {
625         printf("="); Sleep(70);
626     }
627     printf("%%||");
628     //LOADING SCREEN
629
630     //ACTUAL DISPLAY
631     system("cls");
632     displayTime();
633     char *filename = "History.txt";
634     FILE *fptr = NULL;
635     int movieNum; //These variables takes in data read from the history file
636     int NumofSeat;
637     int date;
638     int month;
639     int year;
640     int minute;
641     int hour;
642     char SeatNum[MAX_LEN];
643
644     ////IN CASE OF THE HISTORY FILE BEING EMPTY
645     fptr = fopen(filename, "r");
646     if (NULL != fptr) {
647         fseek(fptr, 0, SEEK_END);
648         int size = ftell(fptr);
649
650         if (0 == size) {
651             printf("No Record Found...\n"); system("pause"); return main();
652         }
653     }
654     ////IN CASE OF THE HISTORY FILE BEING EMPTY
655
656     fptr = fopen(filename, "r");
657     if (fptr == NULL)
658     {
659         fprintf(stderr, "error opening %s\n", filename); return 1;
660     }
661     else {
662         printf("Time of Purchase\tDate of Purchase\tMovie Number\tNumber of
              Seats\t\tSeat Number(s)\n");
```

```
663         printf("\n");
664         while (!feof(fptr)) {
665             fscanf(fptr, "%02d:%02d\t%02d:%02d:%d\t%d\t%d\t%s\n", &hour,
                &minute, &date, &month, &year, &movieNum, &NumofSeat,
                SeatNum);
666             printf("   %02d:%02d\t\t   %02d:%02d:%d\t\t     %d\t\t      %d\t
                \t\t    %s\n", hour, minute, date, month, year, movieNum,
                NumofSeat, SeatNum);
667             Sleep(500);
668         }
669 
670         fclose(fptr);
671         printf("\nData read successfully!\n"); //Signifies that the data was
               read successfully
672     }
673     system("pause");
674     return main();
675     //ACTUAL DISPLAY
676 }
677 
678 void writeData(int movieNum, int NumofSeat, char SeatNum[]) {
679     char *filename = "History.txt";
680     FILE *fptr = NULL;
681     if ((fptr = fopen(filename, "a")) == NULL)
682     {
683         printf("error opening %s\n", filename); return 1; //In case of file
              opening failure
684     }
685     else {
686         fprintf(fptr, "\t%d\t%d\t%s", movieNum, NumofSeat, SeatNum); //
              Writes data into history
687         fclose(fptr);
688         printf("\nData written successfully!\n"); //Signifies data being
              written successfully
689     }
690 }
691 
692 void writeTime() {
693     char *filename = "History.txt";
694     FILE *fptr = NULL;
695     struct tm *local, *gm;
696     time_t t;
697     t = time(NULL);
698     local = localtime(&t);
699     if ((fptr = fopen(filename, "a")) == NULL)
700     {
701         printf("error opening %s\n", filename); return 1; //In case of file
              opening failure
702     }
703     else {
704         fprintf(fptr, "\n%02d:%02d\t%02d:%02d:%d", local->tm_hour, local-
              >tm_min, local->tm_mday, local->tm_mon, local->tm_year + 1900); //
              Writes time of purchase into history
```

```
705           fclose(fptr);
706       }
707   }
```