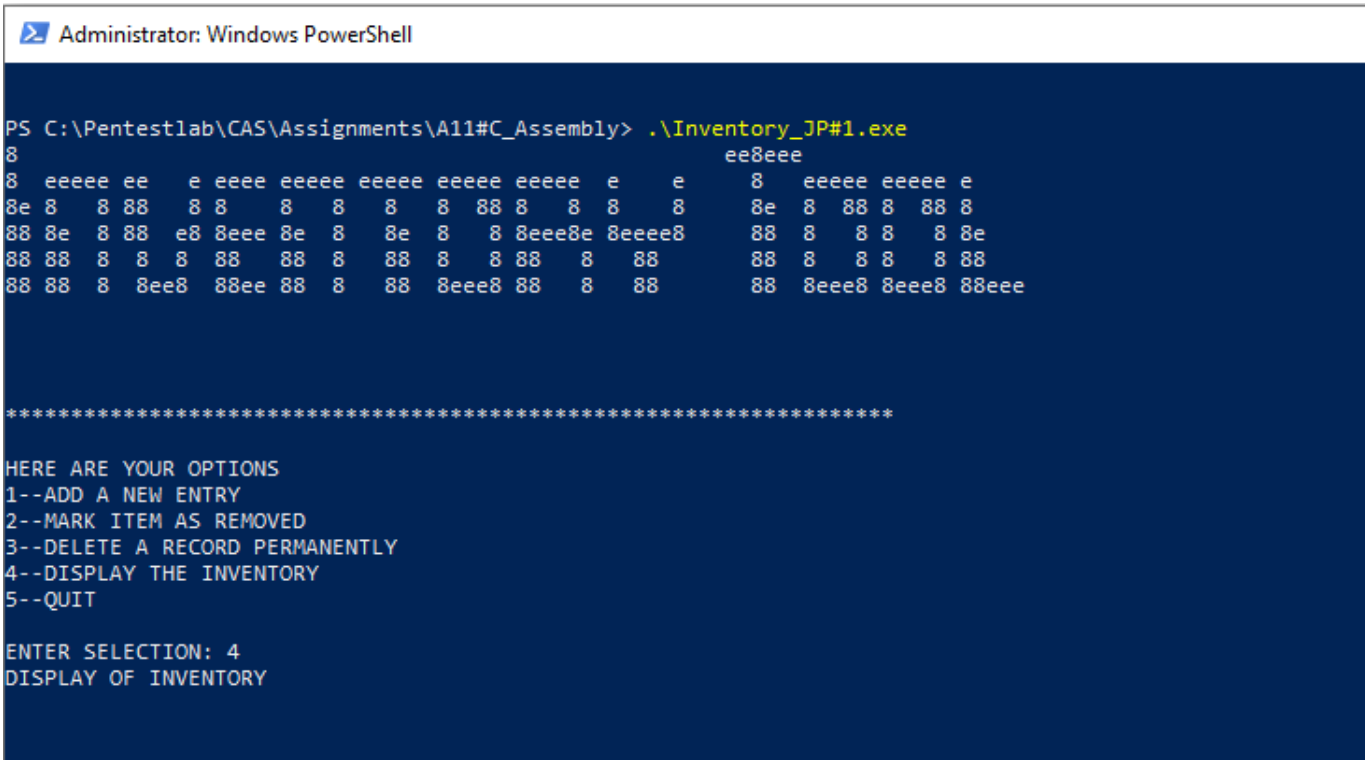
Assignment Series #A11 - Journeyman's Piece

Part 1 - Write inventory tool in C



> Remember our journeyman's piece from the Bash chapter? Let's create something very similar, but without persistence to a file. Instead, we're going to use a two-dimensional array to store the information we require, and we're going to keep the program running, waiting for user input.

You can use the below method currentDate to get the current timestamp:

```c
#include <time.h>
#include <stdio.h>
char* currentDate(char * s) {
   time_t t = time(NULL);
   struct tm *tm = localtime(&t);
   strftime(s, 25, "%c", tm);
   return s;
}

int main() {
   char s[25];
   currentDate(s);
   printf("%s\n", s);
}
```

Requirements

Here is the basic functionality required:

• show our inventory
• add an item to the inventory
• remove items from the inventory

Here are the detailed (simplified) requirements:

• each item can have the following properties
– ID (mandatory)
– date added (mandatory)
– name (mandatory)
– notes - up to 100 characters
– date removed

• adding an item
– reads item properties “name” and “notes” from the command line and
– generates “date added”
– adds the new item to the inventory

• removing an item by ID
– sets the date it was removed
– keeps it from being shown in the inventory listing

• the inventory listing shows the following columns
– ID
– type
– name
– date added
– notes
– date removed (if you implement functionality to show removed items)

The code

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>
#include <time.h>
#define SIZE 75
#define MAX 100

  typedef struct {
    char name[SIZE]; //item name
    char category[SIZE]; //item category
    int id; // item id
    char notes[MAX]; //item notes
    char date_added[25]; //date added
```

```
    char date_removed[25]; //date removed
    } item;

//prompts the user to get a selection
int Menu(void);

//display the options to the user
void DisplayOptions(void);

//function to add a new entry to the inventory
void AddNewEntry(item entry[], int *size);

//function to delete a selected entry from inventory
void Delete(item entry[], int *size, int location);

//function to mark a item as removed
void MarkAsRemoved(item entry[], int *size, int location);

//display the current inventory to screen
void Display(item entry[], int size);

//clears out the entire inventory
void Clear(int *size);

//find location of entry that is going to be edited or deleted
int FindLocation(item entry[], int size);

//fuction for date
char* currentDate ( char *s);

int main()
{
    srand(time(NULL)); // random number generator initalization
    int selection;
    item entry [150];
    int size=0;
    char trash;
    int choice;
    int location;

    DisplayOptions();

    selection= Menu();

    while(selection != 8)
    {
          if (selection==1)
          {
              printf("ADD ENTRIES\n\n");
              //display size of inventory before adding entries
              printf("\nSIZE BEFORE ADDING: %d\n", size);

              AddNewEntry(entry, &size);
```

```
            //display size of inventory after adding entries
            printf("\nSIZE AFTER ADDING: %d\n", size);
        }
           else if(selection==2)
        {
            printf("MarkAsRemoved\n\n");
            //get location of entry to be deleted

            location = FindLocation(entry,size);
            printf("\nLOCATION: %d\n", location);

            MarkAsRemoved(entry, &size, location);
        }

        else if(selection==3)
        {
            printf("DELETE\n\n");
            //get location of entry to be deleted

            location = FindLocation(entry,size);
            printf("\nLOCATION: %d\n", location);

            Delete(entry, &size, location);
        }
        else if(selection==4)
        {
            printf("DISPLAY OF INVENTORY\n\n");
            Display(entry, size);
        }
        else if(selection==5)
        {
            printf("EXIT PROGRAM\n\n");
            exit(0);
        }
        else
        {
            printf("COMMAND NOT RECOGNIZED\n\n");
        }

        printf("\n\n");
        DisplayOptions();

        //clears buffer
        scanf("%c", &trash);

        selection = Menu();
    }

    if (selection==5)
    {
        printf("\nHAVE A NICE DAY!!!\n\n");
    }
    return 0;
}
```

```
char* currentDate ( char *s) {
    time_t t = time ( NULL );
    struct tm *tm = localtime (&t);
    strftime (s, 25, "%c", tm);
    return s;
 }


//display the options to the user
void DisplayOptions(void)
{
    printf("8                                                    ee8eee
\n");
    printf("8  eeeee ee   e eeee eeeee eeeee eeeee eeeee  e    e     8   eeeee
eeeee e \n");
    printf("8e 8   8 88   8 8    8   8   8   8  88 8   8  8    8     8e  8  88 8
88 8 \n");
    printf("88 8e  8 88  e8 8eee 8e  8   8e  8   8 8eee8e 8eeee8     88  8   8 8
8 8e \n");
    printf("88 88  8  8  8  88   88  8   88  8   8 88   8   88       88  8   8 8
8 88 \n");
    printf("88 88  8  8ee8  88ee 88  8   88  8eee8 88   8   88       88  8eee8
8eee8 88eee \n");
    printf("                                                                 \n");
    printf("                                                                 \n");
    printf("\n");
    printf("\n");

printf("**********************************************************************\n\n")
;
    printf("HERE ARE YOUR OPTIONS\n");
    printf("1--ADD A NEW ENTRY\n");
    printf("2--MARK ITEM AS REMOVED\n");
    printf("3--DELETE A RECORD PERMANENTLY\n");
    printf("4--DISPLAY THE INVENTORY\n");
    printf("5--QUIT\n");
}

//prompts the user to get a selection
int Menu(void)
{
    int selection;
    printf("\nENTER SELECTION: ");
    scanf("%d", &selection);

    return selection;
}

//function to add a new enty to the inventory
void AddNewEntry(item entry[], int *size)
{
    printf("\nENTER ITEM:\t\t");
    scanf("%s", entry[*size].name);
```

```
    printf("ENTER Category:\t");
    scanf("%s", entry[*size].category);

    printf("Notes:\t\t");
    scanf("%s", &entry[*size].notes);

    entry[*size].id=rand();

    currentDate(entry[*size].date_added);

    *size = *size + 1;
}

//function to mark item as removed
void MarkAsRemoved(item entry[], int *size, int location) {
    currentDate(entry[location].date_removed);
}

//function to delete a selected entry from inventory
void Delete(item entry[], int *size, int location)
{
    entry[location] = entry[*size - 1];
    *size = *size - 1;
}

//display the current inventory onto the screen
void Display(item entry[], int size)
{
    int i;
    for(i=0; i<size; i++)
    {
        printf("\n");
        printf("ITEM:\t\t %s\n", entry[i].name);
        printf("CATEGORY:\t %s\n", entry[i].category);
        printf("NOTES:\t\t %s\n",entry[i].notes);
        printf("ID:\t %d\n", entry[i].id);
        printf("DATE ADDED:\t\t %s\n", entry[i].date_added);
        printf("DATE REMOVED:\t %s\n", entry[i].date_removed);
    }
}

//find location of entry that is going to be edited or deleted
int FindLocation(item entry[], int size)
{
    int j;
    int userItem;

    //enter the item to delete
    printf("ENTER ITEM ID: ");
    scanf("%d", &userItem);

    for(j=0; j<size; j++)
    {
```

```
            if(entry[j].id==userItem)
            {
                return j;
            }
        }
        return -1;
    }
```