Assignment HW7
Due date (on or before): Announced in class.

1. Textbook, page 246, Exercise 6.21 (5 points)

6.21 *(Airline Reservations System)* A small airline has just purchased a computer for its new automated reservations system. The president has asked you to program the new system. You'll write a program to assign seats on each flight of the airline's only plane (capacity: 10 seats).

Your program should display the following menu of alternatives:

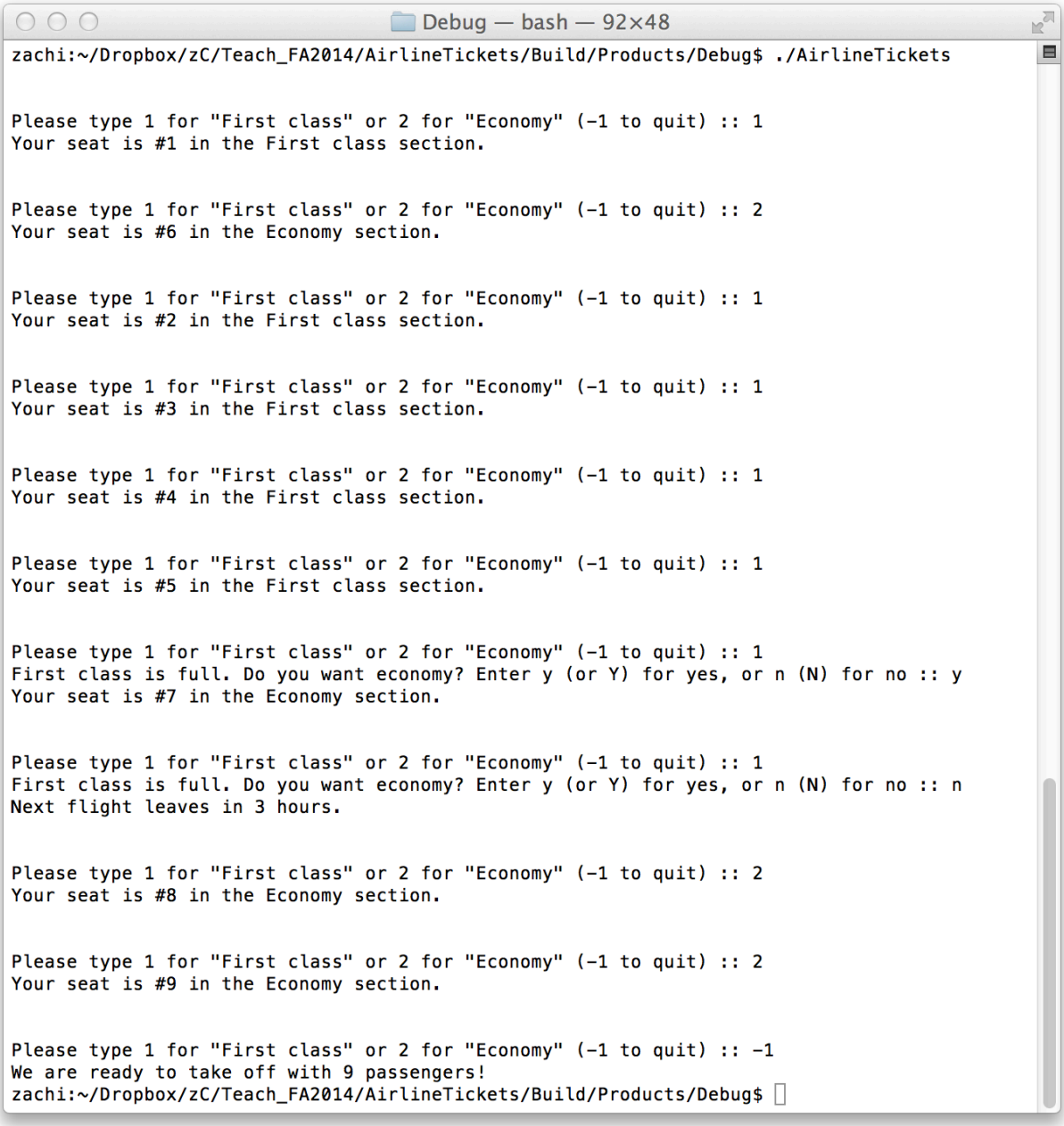
```
Please type 1 for "first class"
Please type 2 for "economy"
```

If the person types 1, then your program should assign a seat in the first class section (seats 1–5). If the person types 2, then your program should assign a seat in the economy section (seats 6–10). Your program should then print a boarding pass indicating the person's seat number and whether it's in the first class or economy section of the plane.

Use a single-subscripted array to represent the seating chart of the plane. Initialize all the elements of the array to 0 to indicate that all seats are empty. As each seat is assigned, set the corresponding element of the array to 1 to indicate that the seat is no longer available.

Your program should, of course, never assign a seat that has already been assigned. When the first class section is full, your program should ask the person if it's acceptable to be placed in the economy section (and vice versa). If yes, then make the appropriate seat assignment. If no, then print the message `"Next flight leaves in 3 hours."`

Screen shot below:



********** Start of code + Screen shot**********

********** End of code + Screen shot**********

Solution 1

```
//
```

```c
//
//  main.c
//  AirlineTickets
// Based on program by An.


#include <stdio.h>

#define CLASS1 5     // seats in the first class
section
#define CLASS2 5     // seats in the economy section


void printPass(int seatnum)
{
    char firstclass[] = "First class section";
    char economyclass[] = "Economy section";

    printf("Your seat is #%d in the %s.\n", seatnum,
           seatnum > CLASS1 ? economyclass :
firstclass);
}


int main(void)
{
    int userInput = 0;
    char response[15];
    int idx1 = 0;         // index of first class
section
    int idx2 = 0;         // index of economy section
    int ii;

    // seats on the plane is first class section +
economy section
    int seats[CLASS1 + CLASS2] ;
    int done = 1;         // 1 - done. 0 - not done
```

```
    for (ii=0; ii < CLASS1+CLASS2 ; ++ii)
        seats[ii] = 0;

    while (userInput != -1 && (idx1 < CLASS1 || idx2 <
CLASS2)){
        if (done == 1){
            printf("\n\nPlease type 1 for \"First
class\" or 2 for \"Economy\" (-1 to quit) :: ");
            scanf("%d", &userInput);
        }

        if (userInput == 1){
            if (idx1 < CLASS1){       // if first class
is still open
                seats[idx1] = 1;      // mark the array
element takend
                idx1++;           // move to the next
available seat
                printPass(idx1);      // print board
pass for the customer
                done = 1;         // ready to read
another request
            }
            else{         // first class is full
                printf("First class is full. Do you
want economy? Enter y (or Y) for yes, or n (N) for no
:: ");
                scanf("%s", &response);

                if (response[0] == 'y' || response[0]
== 'Y'){
                    userInput = 2;// user is ok with
economy seat
                    done = 0; // not ready for a new
request
                    // loop back, re-process the
request
                }
                else{
```

```
                    printf("Next flight leaves in 3
hours.\n");
                    done = 1;
                    continue; // skip the rest of the
code
                }
            }
        }

        if (userInput == 2){
            if (idx2 < CLASS2){
                seats[CLASS1+idx2] = 1;
                idx2++;
                printPass(CLASS1+idx2);
                done = 1;
            } else{           // first class is full
                printf("Economy is full. Do you want
First class? Enter y (or Y) for yes, or n (N) for no
::");
                scanf("%s", &response);

                if (response[0] == 'y' || response[0]
== 'Y'){
                    userInput = 1;
                    done = 0;
                } else
                {
                    printf("Next flight leaves in 3
hours.\n");
                    done = 1;
                }
            }
        }
    }

    if ( idx1== CLASS1 && idx2 == CLASS2)
        printf("Full Flight!!\n");
    else
        printf("We are ready to take off with %d
```

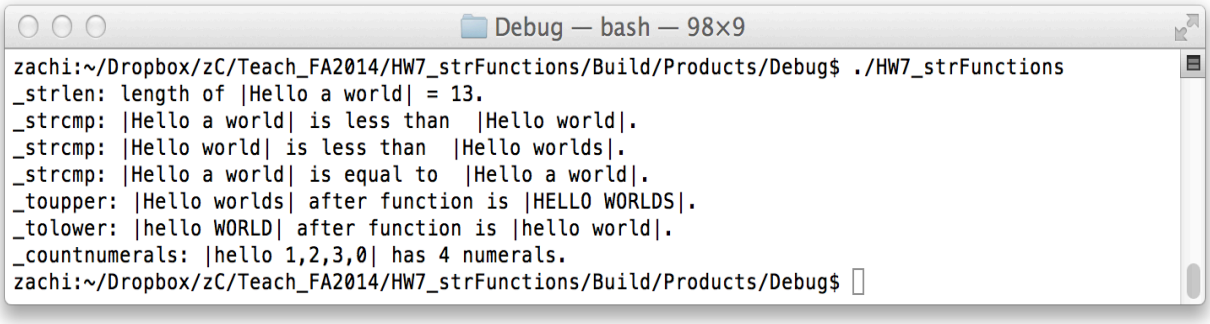
```
passengers!\n",idx1+idx2);

    return 0;
}
```

2. String operations

Create a new project, and copy the below code into your main.c file.
Implement the below functions declared in the skeleton code.

The results should look like the below screenshot (please attach yours!!)



********** Start of code + Screen shot**********

********** End of code + Screen shot**********

```
/**** Start of code to complete ****/
//
//  main.c
//  HW7_strFunctions
//
//

#include <stdio.h>

// _strlen: length of a string
int _strlen(char s[]);

// _strcmp: returns negative, zero, or positive if s is
less, equal, or greater than t (lexicographicaly)
int _strcmp(char s[], char t[]);
```

```
// _toupper: returns the string with all lower cases
converted to upper, and as int the number of changes
void _toupper(char s[]);

// _tolower: you guess...
void _tolower(char s[]);

// _countnumerals: returns how many numerals are in the
string.
int _countnumerals(char s[]);

//
// testing code!!
// Don't change anything in the below code.
char* printcomp(int n)
{
    char* pstr=NULL;
    if ( n==0 )
        pstr = "equals to";
    if ( n<0 )
        pstr = "less than";
    if ( n >0 )
        pstr = "greater than";

    return (pstr);
}

int main()
{

    char str0[80]= "Hello world";
    char str1[80]= "Hello a world";
    char str2[80]= "Hello worlds";
    char str3[80]= "hello WORLD";
    char str4[80]= "hello 1,2,3,0";

    printf("_strlen: length of |%s| =
%d.\n",str1,_strlen(str1));
```

```
    printf("_strcmp: |%s| is %s  |%s|.\n",str1,
printcomp(_strcmp(str1, str0)) ,str0);
    printf("_strcmp: |%s| is %s  |%s|.\n",str0,
printcomp(_strcmp(str0, str2)) ,str2);
    printf("_strcmp: |%s| is %s  |%s|.\n",str1,
printcomp(_strcmp(str1, str1)) ,str1);


    printf("_toupper: |%s| after function is ",str2);
    _toupper(str2);
    printf("|%s|.\n",str2);

    printf("_tolower: |%s| after function is ",str3);
    _tolower(str3);
    printf("|%s|.\n",str3);


    printf("_countnumerals: |%s| has %d
numerals.\n",str4,_countnumerals(str4));


    return 0;
}
/**** End of code to complete ****/
```

Solution 2

```
//
//  main.c
//  HW7_strFunctions
//
//

#include <stdio.h>
```

```c
// _strlen: length of a string
int _strlen(char s[]);

// _strcmp: returns negative, zero, or positive if s is
less, equal, or greater than t (lexicographicaly)
int _strcmp(char s[], char t[]);

// _toupper: returns the string with all lower cases
converted to upper, and as int the number of changes
void _toupper(char s[]);

// _tolower: you guess...
void _tolower(char s[]);

// _countnumerals: returns how many numerals are in the
string.
int _countnumerals(char s[]);

char* printcomp(int n)
{
    char* pstr=NULL;
    if ( n==0 )
        pstr = "equals to";
    if ( n<0 )
        pstr = "less than";
    if ( n >0 )
        pstr = "greater than";

    return (pstr);
}

int main()
{

    char str0[80]= "Hello world";
    char str1[80]= "Hello a world";
    char str2[80]= "Hello worlds";
    char str3[80]= "hello WORLD";
    char str4[80]= "hello 1,2,3,0";
```

```
    printf("_strlen: length of |%s| =
%d.\n",str1,_strlen(str1));


    printf("_strcmp: |%s| is %s  |%s|.\n",str1,
printcomp(_strcmp(str1, str0)) ,str0);
    printf("_strcmp: |%s| is %s  |%s|.\n",str0,
printcomp(_strcmp(str0, str2)) ,str2);
    printf("_strcmp: |%s| is %s  |%s|.\n",str1,
printcomp(_strcmp(str1, str1)) ,str1);


    printf("_toupper: |%s| after function is ",str2);
    _toupper(str2);
    printf("|%s|.\n",str2);

    printf("_tolower: |%s| after function is ",str3);
    _tolower(str3);
    printf("|%s|.\n",str3);


    printf("_countnumerals: |%s| has %d
numerals.\n",str4,_countnumerals(str4));


    return 0;
}


// _strlen: length of a string
int _strlen(char s[])
{
    int ii=0;
    while (s[ii++]);
    return (ii-1);
}
```

```c
// _strcmp: returns negative, zero, or positive if s is
less, equal, or greater than t (lexicographicaly)
int _strcmp(char s[], char t[])
{
    int ii;

    for (ii=0; s[ii]==t[ii]; ii++)
        if (s[ii] == '\0')
            return (0);

    return (s[ii]-t[ii]);

}

// _toupper: returns the string with all lower cases
converted to upper
void _toupper(char s[])
{
    int ii=0;

    while (s[ii]){
        if (s[ii]>='a' && s[ii]<='z')
            s[ii] += 'A'-'a';
        ++ii;
    }

}

// _tolower: you guess...
void _tolower(char s[])
{
    int ii=0;

    while (s[ii]){
        if (s[ii]>='A' && s[ii]<='Z')
            s[ii] += 'a'-'A';
        ++ii;
    }
```

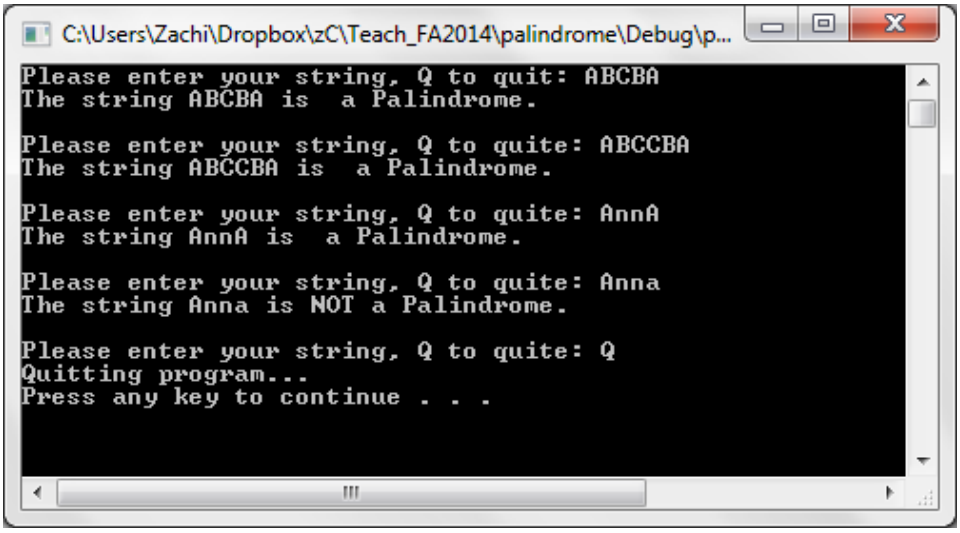
```
}
// _countnumerals: returns how many numerals are in the
string.
int _countnumerals(char s[])
{
    int ii=0;
    int cnt=0;

    while (s[ii]){
        if (s[ii]>='0' && s[ii]<='9')
            cnt++;
        ++ii;
    }
    return (cnt);
}
```

3. Palindrome
(similar to Textbook, page 251, Exercise 6.31)

Write a program that determines whether a string is a palindrome. See example screenshot below. The program will ask the user for an input string (assume not longer than 79 characters). The program will then print whether the string is a palindrome or not, and prompt the user to input new string. The process will stop when the user inputs a string of one character, Q (if entered at the first time, no palindrome is checked).

What is a palindrome? A string that spells the same forward and backward.
Example:
ABCBA is a palindrome
ABCCBA is a palindrome
AnnA is a palindrome
ABCD is NOT a palindrome
Anna is NOT a palindrome (we are case sensitive here)



********** Start of code + Screen shot**********

********** End of code + Screen shot**********

/* the below can serve as a hint if needed */

```
/**** Start of code to complete ****/
#include <stdio.h>
#include <stdlib.h>

// Returns length of string
int strlength(char* s)
{
  // Assume s contains proper '\0' terminated string

/* Fill in your code here */

}

// Returns true for palindrome, false if not
int isPalindrome(char* s)
{

/* Fill in your code here */


}

// main() function
int main(void)
{

  char str[80];
  int b;
```

```
  printf("Please enter your string, Q to quit: ");
  scanf("%s",str);

  while (/* Fill in your code here */)
  {

       b = isPalindrome(str);

              /* Fill in your code here */

  }

  printf("Quitting program...\n");

  system("Pause");
  return(0);
}

/**** End of code to complete ****/
```

Solution 3

```
#include <stdio.h>
#include <stdlib.h>

// Returns length of string
int strlength(char* s)
{
  // Assume s contains proper '\0' terminated string
  char* p = s;
  while (*p!= '\0') p++;
  return (p-s);
}

// Returns true for palindrome, false if not
int isPalindrome(char* s)
{
  int l = strlength(s);
  int ii = 0;
  int mid = l/2+1;

  while ( (ii < mid)  && s[ii]==s[l-1-ii] )
       ++ii;

  return (ii>=mid);
}

// main() function
int main(void)
{

  char str[80];
  int b;
```

```
  printf("Please enter your string, Q to quit: ");
  scanf("%s",str);

  while (!(str[0]=='Q'  && str[1]=='\0'))
  {

       b = isPalindrome(str);

       printf("The string %s is %s a Palindrome.\n\n",str,(b) ? "": "NOT");

       printf("Please enter your string, Q to quite: ");
       scanf("%s",str);
  }

  printf("Quitting program...\n");

  system("Pause");
  return(0);
}
```