Wasfi Momens

CSC3320 System Level Programming
Homework 4

Due at 11:59 pm on Tuesday, Nov. 22, 2016

Note: All the questions below are from the textbook by Dr. King.

Q1. Write the following function:

```
Bool search (int a[], int n, int key);
```

a is an array to be searched, **n** is the number of elements in the array, and key is the search key. ***search*** should return *TRUE* if key matches some element of **a**; *FALSE* if it doesn't. Use pointer arithmetic – not subscripting – to visit array elements.

```
Bool search (int a[], int n, int key) {

      int f;
      for( f=0; f< n; f++) {
            if (key == *(a+f)) {
                  return TRUE;
            }
      }
      return FALSE;
}
```

Q2. Let f be the following function:

```
int f(char *s, char *t)
{
      char *p1, *p2;
      for(p1 = s ; *p1;p1++){
            for(p2 = t ; *p2;p2++)
                  if (*p1 == *p2 ) break;
            if(*p2 == '\0')  break;
      }
      return p1 – s;
}
```

(a) What is the value of *f("abcd", "babc")*?
(b) What is the value of *f("abcd", "bcd")*?

(c) In general, what value does f return when passed two stings **s** and **t**?

a) 3
b) 0
c) this function searches for each character in t to see if it is also in s and then returns then either breaks if all characters are found in both char pointers or breaks at the index of the char where it has found a character in t that is not in s.

Q3. What will be the value of the string str after the following statements have been executed? And explain why?

```
strcpy(str, “tire-bounchon”);
strcpy(&str[4], “d-or-wi”);
strcat(str,”red?”);
```

str will equal “tired-or-wired?\0”. Strcpy copies the second parameter (a string) to the first parameter (a string), but if the parameter is an address then it will copy the second string (or address of a string) to the location of the specified address In str. So at first str will be “tire-bounchon\0”, then it will be “tired-or-wi\0”, and then “tired-or-wired\0”.

Q4. The following function supposedly creates an identical copy of a string. What’s wrong with the function?

```
char *strdup(const char *p)
{
     char *q;
     strcpy(q,p);
     return q;
}
```

q is a pointer that does not point to anything, so we need to point it to memory large enough to match the size of char p and the null terminator.

Q5. a) Declare a tag for an enumeration whose values represent the twelve months of the year.

```
enum month {
      january = 1,
     february = 2,
     march =3,
     april = 4,
     may = 5,
     june = 6,
     july = 7,
```

august = 8,
september = 9,
october = 10,
november = 11.
december = 12
};

b) Use ***typedef*** to define a name for enumeration of part (a).
typedef enum month MONTH;

Q6. The following structures are designed to store information about objects on a graphics screen. A **point** structure stores the x and y coordinates of a point on the screen. A **rectangle** structure stores the coordinates of the upper left and lower right corners of a rectangle.

```
struct point { int x, int y;};
struct rectangle { struct point upper_left, lower_right; };
```

1) Write functions that perform the following operations on a rectangle structure r passed as an argument:

(a) Compute the center of r, returning it as point value. *Function name:* ***Center***

```
struct point center (struct rectangle r) {

        /* center of rectangle is ((upper_left x + lower_right x)/2) + ((upper_left y + lower_right y)). This is because the coordinates of the upper left point will be complement of the lower right corner. */

        int x_add= (r.upper_left.x  + r.lower_right_x) / 2;
        int y_add = (r.upper_left.y + r.lower_right_y) /2;

        struct point centerrec;
        centerrec.x = x_add;
        centerrec.y = y_add;
        return centerrec;
}
```

(b) Move r by x units in the x direction and y units in the y direction, returning the modified version of r. (x and y are additional arguments to the function.) *Function name:* ***Move***

```
struct rectangle move (struct rectangle r, int x, int y) {
        r.upper_left_x += x;
        r.upper_left.y += y;
        r.lower_right.x += x;
```

```
        r.lower_right.y += y;
        return r;
}
```

(c) Determine whether a point p lies within r, return TRUE or FALSE. (p is an additional argument of type struct **point**). *Function name:* ***IsInRectangle***

```
/*  assuming #include<stdbool.h> */
bool isinrec (struct point p, struct rectangle r){
    if(
}
```

2) Suppose following declaration is in effect.

```
struct rectangle *p;
```

Assume that we want p point to a rectangle structure whose upper left corner is at *(2,3)* and whose lower right corner is *(0,1)*. Write a series of statement that allocates such a structure and initialize it as indicated.

3) Now modify **rectangle** structure by adding one more member in it as following:

```
struct rectangle
{ struct point upper_left, lower_right;
  char *name;
};
```

Assume that we want p point to a rectangle structure whose upper left corner is at *(2,3)* ,whose lower right corner is *(0,1)* and name is "*MyRect*". Write a series of statement that allocates such a structure and initialize it as indicated.

```
rectangle rec;
rec.upper_left.x = 2;
rec.upper_left.y = 3;
rec.lower_right.x = 0;
rec.lower_right.y=1;
rec.name = "MyRect";
```

4) Write a series of statements to define a rectangle array rectArr with 10 elements, allocate memory to this array and initialize the upper left corner, lower right corner and name as (0,0), (1,1) and "Rect" respectively for each element in this array.

Submssion:

- Upload an electronic copy (MS word or pdf) of your answer sheet to the folder named "**HW4**" of the dropbox in the iCollege system
- Please add the homework number and your name at the top of your answer sheet.
- Name your file in the format of HW4_FirstnameLastname (eg. HW4_YuanLong.docx, HW4_YuanLong.pdf)