Universidad de Puerto Rico en Bayamón

Departamento de Ciencias de Computadoras

COTI 4039 – Comparación de Lenguajes de Programación

Asignación #2 – 60 puntos

1. (15 puntos) Escriba un programa en C (rock\_paper\_scissors.c) que use funciones para implementar el juego de Piedra, Papel y Tijera. En este programa el usuario jugará contra la computadora. A continuación se muestra un ejemplo de su ejecución:

Enter your selection (0=Rock, 1=Paper, 2=Scissors): ***0***

My selection was 2=Scissors. You won!

Dou you want to play again (y/n)? ***y***

Enter your selection (0=Rock, 1=Paper, 2=Scissors): ***0***

My selection was 1=Paper. You lost!

Dou you want to play again (y/n)? ***y***

Enter your selection (0=Rock, 1=Paper, 2=Scissors): ***2***

My selection was 1=Paper. You won!

Do you want to play again (y/n)? ***n***

Thanks for playing. You won 66.7% of the games.

El programa computará al azar la jugada de la computadora (0 = Piedra, 1 = Papel, 2 = Tijera), le pedirá al usuario su jugada y determinará quién ganó. Se deberá continuar jugando mientras el usuario así lo desee. Por último, el programa mostrará el porciento de juegos ganados a un lugar decimal, asumiendo que los empates no contarán como juegos válidos. Valide la selección del usuario y la confirmación para continuar el juego.

Recuerde que:

1. La piedra le gana a la tijera pero pierde con el papel.
2. El papel le gana a la piedra pero pierde con la tijera.
3. La tijera le gana al papel pero pierde con la piedra.

**Pista:** Las funciones srand y rand en <stdlib.h> le servirán para generar números aleatorios.

2. (10 puntos) Escriba un programa en C (fibonacci\_calc.c) que calcule el enésimo término de la sucesión de Fibonacci. A continuación se muestra un ejemplo de su ejecución:

Enter a non-negative integer: ***7***

Using iteration, the 7-th number in the Fibonacci sequence is 13

Using recursion, the 7-th number in the Fibonacci sequence is 13

Thanks for using this program

La sucesión de Fibonacci es la siguiente:

0, 1, 1, 2, 3, 5, 8, 13, …

Los elementos en las posiciones 0 y 1 son siempre 0 y 1, respectivamente. Cada elemento en la posición n, para n > 1, se obtiene sumando los dos elementos anteriores. Esto quiere decir que:

* *fibo(0) = 0*
* *fibo(1) = 1*
* *fibo(n) = fibo(n – 1) + fibo(n – 2).*

El programa deberá determinar el n-ésimo elemento de la sucesión de Fibonacci utilizando iteración y recursión. Si el número es menor de 0, el programa deberá mostrar un mensaje de error y terminar. La versión que usa recursión regular puede explotar si se tratan valores mucho mayores de 30.

3. (15 puntos) Escriba un programa en C (array\_stats.c) que pida los elementos para un arreglo de 10 números enteros y que determine el elemento mínimo, el promedio de los elementos y la cantidad de elementos impares. A continuación se muestra un ejemplo de su ejecución:

Enter the 10 elements: 5 4 3 2 1 6 -7 8 9 10

The minimum element is -7

The average of the elements is 4.1

There are 5 odd elements

Thanks for using this program

Dos de las funciones que manejan el arreglo deberán utilizar la notación de apuntadores. El promedio se deberá mostrar a un lugar decimal.

4. (10 puntos) Modifique el programa de manejo de arreglos en C discutido en clase (my\_array\_fns.c) para que implemente el algoritmo de búsqueda secuencial. El prototipo de la función de búsqueda será el siguiente:

int linear\_search(int arr[], int len);

También defina una función que implemente el algoritmo de ordenamiento por selección de forma tal que permita ordenar tanto ascendente como descendentemente. El prototipo de la función de ordenamiento será el siguiente:

void selection\_sort(int arr[], int len,

int (\*compare)(int val1, int val2));

5. (10 puntos) Modifique el programa de manejo de arreglos en C discutido en clase (my\_string\_fns.c) para que implemente una función que convierta una cadena de caracteres a su equivalente en mayúscula. El prototipo de la función será el siguiente:

void strtoupper(char \*str);

También defina una función que reciba una cadena de caracteres y devuelva su equivalente como un entero, sea éste un número negativo, cero o positivo. El prototipo de la función será el siguiente:

int strtoint(const char \*str);

Si la cadena no es numérica, la función devolverá cero.

**Pistas:** La función toupper en <ctype.h> le servirá para convertir a mayúscula un carácter a la vez. La función isdigit en <ctype.h> le servirá para determinar si un carácter es un dígito.

El siguiente algoritmo le permite obtener el equivalente como entero de una cadena de caracteres:

*Initialize an accumulator to 0*

*For each character in string:*

*Get the equivalent digit for the character*

*Multiply the digit by 10*

*Add this result to the accumulator*

Recuerde que si a un carácter numérico le resta el código de ‘0’ obtendrá el equivalente del carácter como dígito.

**Nota:**

Documente sus programas colocando su nombre, número de estudiante, fecha de creación, el nombre del archivo fuente y una descripción general. Cada función y cada fragmento importante de su programa deberá contener un comentario descriptivo.