Computación I, Facultad de Ingeniería, UNLPam, ciclo lectivo 2022

Primera Evaluación Parcial, octubre

Consigna Preliminar

La evaluación práctica de programación será sobre un programa que controle un hipotético entorno físico de producción, que necesita mantener la temperatura relativamente estable, dentro de un rango permitido. Se deben emplear para lograr el objetivo dispositivos de enfriamiento o calefacción, según se necesite.

Se solicita en forma preliminar escribir el código C de un "bastidor" o marco de un programa que incluya al menos 4 (cuatro) funciones que permitan :

1) la lectura de temperaturas, entre -25° y +50°, retornando, un entero en ese rango, ver más adelante propuesta de *int sensa_temp()*

2) templar el ambiente, mediante una función calefactor(parm1, parm2) que debe incrementar y retornar la temperatura recibida en 1°C, excepto cuando debe ser acelerado el calefaccionado, donde el incremento debe ser en 5°. La indicación del acelerado debe ser considerando un parámetro adicional.

3) enfriar el ambiente, mediante una función enfriador(parm1, parm2) que debe bajar y retornar la temperatura recibida en 1°C, excepto cuando debe ser acelerado el enfriamiento, donde el descenso debe ser en 5°. La indicación del acelerado debe ser considerando un parámetro adicional.

4) calcular el promedio simple de las temperaturas registradas, mediante una función promedio(parm1, parm2), que recibe la cantidad de lecturas realizadas y un acumulado de los valores leídos y, como es de esperar, devuelve el promedio calculado. Aquí debe usarse aritmética entera, de modo que el promedio debe ser siempre expresado como un entero.

El programa deberá incluir la declaración que permita el acceso a las funciones de las bibliotecas <stdlib.h>, <stdio.h>, <math.h>, <time.h> y la declaración de las constantes simbólicas: a)MAXNUM , que podría usarse como la cantidad máxima de lecturas a realizar y analizar, b) MAXTEMP, MINTEMP, PROMTEM (calculada esta última como el cociente sobre 2 de la diferencia entre MAXTEM Y MINTEM), las que que podrían ser usadas como las temperaturas máximas, mínimas y promedio respectivamente, y c) PROM_10MAS y PROM_10MEN que deben computarse como PROMTEM + 10% Y PROMTEM – 10% respectivamente y representan valores superiores e inferiores en un 10% respecto a la temperatura intermedia, considerando los

máximos y mínimos definidos previamente

Requisitos Iniciales

El programa debe admitir una cantidad fija de lecturas de temperatura, que se ingresarán como única entrada por teclado y, cada 3 lecturas, se debe invocar a la función que calcula el promedio de TODAS las temperaturas registradas hasta el momento.

Se invocará a la función de `calefactor()` cuando la temperatura registrada esté debajo de 0° y a `enfriador()` cuando esté por encima de 40°, dichas funciones se invocarán aceleradas cuando se deban invocar en forma consecutivas, por ej. si se invoca `enfriador()` dos veces seguidas, en la segunda llamada debe ser "acelerada".

Al salir del ciclo principal, antes de finalizar, el programa informará cuales fueron el primer y último promedios calculados, cuántas veces se debió acelerar el enfriamiento y cuantas veces debió acelerar la calefacción.

Referencias a la función de registro/lectura de datos.

Permite leer los datos sin tipear cada vez todas las temperaturas

```
int sensa_temp()

{

       //La fórmula general, para la lectura en el intervalo, sería:

      return (rand()%(INTERVALO+1)+MINIMO)

      //esta función rand() entrega un número pseudo al azar y se debe usar

      // ejecutando previamente

      // a su invocación (al principio de main() o donde corresponda)

      //la función complementaria srand(time(NULL)); inicialización o

      // "sembrado" del número pseudo al azar

      //ambas funciones se definen en el encabezado stdlib.h

      /* para mas detalles ver https://www.aprenderaprogramar.com buscando

      "generar numeros al azar en C"*/

}
```

Referencias a programación por eventos, sin lecturas bloqueantes

Permite ejecutar algún bloque de código de acuerdo al tiempo transcurrido, por ej. desde el inicio de ejecución del programa

---

```
//Cálculo de intervalo de tiempo transcurrido en segundos
//cada n segundos se atiende un evento: por ej. lectura de datos del
// "sensor"
//cada evento se debe atender por separado, salvo que tenga el mismo
//requerimiento de tiempo
#include <stdlib.h>

#include <stdio.h>
#include <time.h> //encabezado necesario para cálculo de tiempo

//constantes simbólicas usadas en el código

#define TOTTEMP 20 //Tiempo total de ejecución del programa en segundos

#define TEV1 5 //Tiempo para el Evento 1

//"marco" o "esqueleto" de nuestro programa

int main(){

	clock_t tiempo_inicio, tiempo_final, inicio_evento1, fin_evento1 ;

	//variables para almacenar tiempo

	double segundos_v=0, tiempo_tot =0; //tiempo transcurrido

	//se inicializa el tiempo

	tiempo_inicio = clock();

	inicio_evento1 = clock(); //inicio del conteo de tiempo

	//clock(): función de encabezado time.h, devuelve el tiempo

	//transcurrido desde el inicio del programa

	while  (tiempo_tot< TOTTEMP){ //o do, se ejecuta durante m segundos
	fin_evento1 = clock();

		tiempo_final = clock();

		segundos_v = (double)(fin_evento1 -inicio_evento1)/ CLOCKS_PER_SEC;

		//la expresión (double)(t2 -t1) / CLOCKS_PER_SEC;
```

```
        //permite computar, previa conversión con el cast (double),
        //el tiempo transcurrido, a partir de info
        //en 2 variables (t1, t2)) de tipo "clock_t"
        tiempo_tot= (double)(tiempo_final-tiempo_inicio) / CLOCKS_PER_SEC;
        // CLOCKS_PER_SEC: constante simbólica de encabezado time.h:
        // entrega la frecuencia del reloj, cantidad de pulsos/seg.
        if (segundos_v >=TEV1)
               //transcurrieron 5seg o mas, se procesa el evento
               {inicio_evento1 = fin_evento1;
               // se reinicia el conteo de tiempo!
               //al inicio del ciclo donde se procesa el evento
               // código que hace algo...acumula, compara, etc.
               printf("tiempo entre eventos %8.2lf\n", segundos_v);
               }
    } //fin while o do while
    printf("tiempo transcurrido Total de ejecución %lf", tiempo_tot);
    // aproximadamente
    return (0);

}
```