Preparacion para Examen

Resolución de Problemas (2.5 horas)

Problema 1: El problema de distribución de libros escolares

Imagina que estás ayudando a una escuela a distribuir libros entre las aulas para un nuevo semestre. La escuela tiene una cantidad total fija de libros disponibles, digamos 720 libros. Lo ideal sería que cada aula recibiera la misma cantidad de libros para garantizar una distribución justa entre los estudiantes. Sin embargo, debido a una actualización reciente del software, el sistema de gestión de inventario ha desactivado temporalmente las operaciones de división (/) y módulo (MOD). La tarea es determinar cuántas aulas pueden recibir la misma cantidad de libros, dado que lo ideal sería que cada aula recibiera 30 libros. Además, si no es posible una división perfecta, calcula cuántos libros quedarían sin asignar. Esta información ayudará a la escuela a planificar cualquier compra adicional o espacio de almacenamiento necesario para los libros no utilizados.

Resumen del problema: La escuela necesita distribuir libros de manera que cada aula reciba exactamente 30 libros. Con un total de 720 libros disponibles, cree un método para determinar cuántas aulas completas se pueden abastecer y cuántos libros quedarían sin distribuir si no es posible una división exacta, todo ello sin utilizar operadores de división o módulo.

Casos de prueba

Entrada Total libros)	**Salidas (cursos completos libros sin distribuir)**
720	24 0
745	24 25
150	5 0
98	6 8
-150	Error! debe ser positivo

La salida debe contener en la primera línea el total de cursos con asignación completa y en la segunda la cantidad de libros sobrantes. En caso de una entrada negativa, envía un mensaje de error.

Restricciones de ayuda: En caso de requerirlo, pida ayuda al ayudante o profesor(a), tiene 3 posibilidades para cada problema.

Problema 2: Carga de Choque Trueno de Pikachu

Pikachu se está preparando para una gran batalla, y necesita que le ayudes a gestionar sus niveles de electricidad para asegurarse de que puede lanzar el Choque *Trueno* más fuerte posible. Pikachu carga su energía eléctrica en una batería por cada día de la semana, distribuidas en un arreglo de 7 posiciones. Para estar bien preparado, Pikachu requiere saber cuál es el día de máxima electricidad que puede encontrar en las baterías almacenadas para *lanzar su Golpe Trueno*.

Tu tarea es escribir un programa que:

1. Lea un arreglo de enteros donde cada entero representa la energía eléctrica cargada en una batería, una por cada día de la semana .
2. Identifique la posición y valor de la energía máxima de cualquier batería dentro del arreglo.
3. Muestre la energía máxima y el correspondiente día de la semana en palabras agregando el sufijo " es un buen dia para un Golpe Trueno".

Restricciones

1. El arreglo puede contener números positivos y negativos. Los números negativos representan días de pérdidas de energía respecto de la semana anterior.
2. Los días empiezan el lunes (0 = lunes, 1 = martes, ... 6 = domingo).

Entrada

- Un array de 7 enteros que representan la energía de cada batería diariamente.

Salida

- Un único entero para la energía máxima que Pikachu puede recoger de una batería, y el día correspondiente en palabras con el sufijo " de Golpe Trueno".

Casos de prueba

Entrada	Salida
5 -3 7 2 -8 10 0	10 sábado es un buen dia para un Golpe Trueno
15 -3 7 2 -8 10 -1	15 lunes es un buen dia para un Golpe Trueno
1 1 1 1 1 1 1	1 cualquier día es un buen dia para un Golpe Trueno
-2 -4 -5 -1 -1 -1 -10	nunca es un buen día para un Golpe Trueno

Restricciones de ayuda: En caso de requerirlo, pida ayuda al ayudante o profesor(a), tiene 3 posibilidades para cada problema.

Anexo1: Plantilla ADCP con ejemplo (en caso que la necesite)

Problema: Suma que te suma

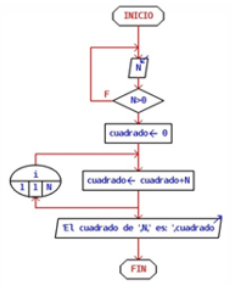
Construya un programa que calcule el cuadrado de un número no negativo usando únicamente sumas.

ANALISIS

Entradas	El límite superior del rango **N**
Proceso	El producto de 2 números (n, m) es posible calcularlo como la sumatoria n – ésima de m. n * m = m + m + m … n veces por ende n * n = n + n + … n veces Se necesita una variable acumuladora para guardar la sumatoria n - ésima de n.
Salida	Cuadrado de un número natural
Restricciones	N > 0

DISEÑO

Diseño del algoritmo



Diseño de casos de prueba

Entrada	Salida
3	9
-2	Error!, debe ser positivo
0	0

CONSTRUCCION

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

int main() {

    float cuadrado,i,n;
    do {
        printf("Ingrese n");
                    scanf("%d",&n);
    } while (n<=0);
    cuadrado=0;
    for (i=1;i<=n;i++) {
        cuadrado=cuadrado+n;
    }
    printf("El cuadrado de %d es %d", n, cuadrado);
            system("pause");
    return 0;
}
```

PRUEBAS

Ejecución de los casos de prueba

Entrada	Salida	Éxito
3	9	☺
-2	Error!, debe ser positivo	☺
0	0	☺