
**Apellido, Nombre:**

.....

**Mail:**

.....

**Padrón:**

.....

| Teórico / Práctico - Entrego.....hojas | | | | |
|---|---|---|---|---|
| 1: ..... | 2: ..... | 3: ..... | 4: ..... | 5: ..... |

| Nota Final |
|---|
| ..... |

## Aclaraciones:

- **Antes de comenzar a resolver el parcial, complete sus datos en esta hoja.**
- **Los ejercicios deben ser implementados en el lenguaje de programación C, respetando las buenas prácticas.**
- **Para cada ejercicio práctico se recomienda realizar un análisis y un diagrama del problema y la solución.**
- **Se deben numerar TODAS las hojas e inicializarlas con nombre, apellido y padrón.**
- **La aprobación del parcial está sujeta a la aprobación de al menos el 60% del mismo.**

## Ejercicios:

**1 - De los elfos.** Lórien es un país habitado por Elfos y es muy difícil entrar sin ser visto por uno de ellos. Consciente de la guerra que se avecina, Celeborn, señor de Lórien, quiere aumentar la seguridad. Para esto, determinó que todo punto del país debe poder ser visto directamente por al menos 1 Elfo.
Las dimensiones de Lórien, permiten verlo como una matriz, cada casillero está representada por una E si hay un elfo, una A si hay un árbol o una N si no hay ni Elfo ni árbol.
Los elfos pueden ver horizontal y verticalmente hasta una distancia de 10 casilleros, pero no pueden ver a través de los árboles ni en diagonal.

a. Determinar si Lórien está seguro, según las nuevas condiciones impuestas por Celeron, cumpliendo con la siguiente firma.

```
bool esta_segura(char lorien[MAX_ALTO][MAX_ANCHO], int alto, int ancho);
```

**2 - De los hombres.** Aragorn, luego de convertirse en Rey, decidió condecorar a quienes participaron activamente en las batallas de los Campos de Pelennor y Cuernavilla y hayan matado al menos 10 Orcos o 2 Uruk-Hai. A ellos les regalará una hectárea, escriturada, con quincho y pileta.
Se tienen entonces, 2 vectores de guerreros ordenados ascendentemente por nombre del guerrero, uno con quienes participaron en la Batalla de los Campos de Pelennor y otro con quienes participaron en la Batalla de Cuernavilla y se necesita obtener un 3er vector con quienes cumplan las condiciones descritas anteriormente.

```
typedef struct guerrero {
        char nombre[MAX_NOMBRE];
        int edad;
        int orcos_matados;
        int uruk_matados;
} guerrero_t;
```

a. Crear un procedimiento que guarde en un vector, quienes son los guerreros merecedores de tan generoso reconocimiento.
b. Para la unión y la diferencia simétrica ¿Cuáles son los topes máximo y mínimo que pueden obtenerse?

*"A lo hecho, pecho..."* - H.J.S.

**3 - De los enanos.** Las manadas se organizan de tal forma que el más lento vaya adelante, para que todo el grupo siga su paso y no lo dejen atrás. De otra forma, los más rápidos se alejarían de los más lentos, haciendo que todos sean vulnerables.
Esta configuración es la que eligió Thorin para que La Compañía emprenda su viaje hacia la Montaña Solitaria.
La velocidad de cada enano está determinada por la velocidad propia del enano menos el peso de su mochila divido 10.
Por ejemplo, si Bofur tiene una velocidad propia de 10 km por hora y lleva en su mochila 5 elementos, uno de 10 kilos, otro de 7 kilos, otro de 8 kilos, otro de 6 kilos y otro de 9 kilos, su velocidad será:
**velocidad = 10 - ((10 + 7 + 8 + 6 + 9)/10) = 10 - 40/10 = 10 - 4 = 6 km/h.**

Se tiene un vector de integrantes de la compañía donde cada integrante posee una mochila con elementos, las estructuras son como las que siguen:

```
typedef struct elemento {
	char descripcion[MAX_DESCRIPCION];
	float peso;
} elemento_t;

typedef struct integrante {
	char nombre[MAX_NOMBRE];
	float velocidad;
	elemento_t mochila[MAX_ELEMENTOS];
	int cantidad_elementos;
} integrante_t;
```

a. Ordenar a los integrantes de la compañía por velocidad, para que ninguno quede rezagado.
b. Un nuevo enano se sumó a la compañía a la altura de Ezeiza, iba en una C90 para ganar velocidad (pero hacía más ruido que otra cosa), crear un procedimiento que lo ingrese al vector ya ordenado, sin perder dicho orden.

**4 - De los hobbits.** Los hobbits suelen tener una vida rutinaria y tranquila, se pasan bastante rato en tabernas tomando cervezas y luego de la 5ta o 6ta les resulta muy fácil divertirse con cualquier cosa. En esta oportunidad pusieron varios vasos de cerveza en una hilera y quieren determinar **recursivamente** si están ordenados por cantidad de cerveza. La hilera de vasos con su contenido puede verse como un vector de floats.

a. Crear una rutina recursiva que resuelva el entrevero, antes de que pase a mayores.
b. Explique: Qué parámetros utiliza y por qué. Cuáles son los componentes esenciales de una rutina recursiva y donde se ven en su solución.

**5 - De los ents.** Los ents son una raza de seres inteligentes parecidos a árboles. La mayor diferencia respecto a los árboles normales es que ellos pueden desplazarse. Su principal función era que cuidaran de los otros árboles de los peligros que acechan. Habitaban en todos los bosques del mundo, pero al final de la Tercera Edad solo quedaban unos pocos en el bosque de Fangorn.

a. Crear una estructura que permita representar lo mejor posible a un ent, para tener un registro de los mismos, llevar un control y evitar su extinción. #losentstambienhacenlafotosintesis

En este punto es importante recordar, que los ents son seres ficticios, por lo que, por favor, use su imaginación y no se limite a unos pocos campos, puede crear estructuras dentro de la estructura principal, por ejemplo. Piense en qué cosas podría hacer un ent o para qué puede servir un ent.
Les dejamos algunas ayudas, "algunos podrían servir, una vez deforestados, para unos ricos asaditos", "cada ent quizás tenga que cuidar de ciertos árboles", "puede ser que los ents tengan políticos, intendentes, reyes".

1) Pedia determinar si el lugar era seguro, determinar si todos lso casilleros de una matriz cumplen con algo.
Una forma es asumir que el lugar es seguro hasta determinar lo contrario, es decir, afirmo que es seguro,
a menos que encuentre 1 casillero que no lo sea.

```
bool casillero_seguro(char lorien[MAX_ALTO][MAX_ANCHO], int ancho, int alto,
int i, int j){
	bool es_seguro = false;

	if (lorien[i][j] == 'E') es_seguro = true;

	int contador = 1;
	while(contador <= 10 && lorien[i+contador][j] != 'A' &&
		 !es_seguro && (i+contador) < alto){
		if (lorien[i+contador][j] == 'E') es_seguro = true;
		contador++;
	}

	contador = 1;
	while(contador <= 10 && lorien[i-contador][j] != 'A' &&
		!es_seguro && (i-contador) >= 0){
		if (lorien[i-contador][j] == 'E') es_seguro = true;
		contador++;
	}

	contador = 1;
	while(contador <= 10 && lorien[i][j+contador] != 'A' &&
		!es_seguro && (j+contador) < ancho){
		if (lorien[i][j+contador] == 'E') es_seguro = true;
		contador++;
	}

	contador = 1;
	while(contador <= 10 && lorien[i][j-contador] != 'A' &&
		!es_seguro && (j-contador) >= 0){
		if (lorien[i][j-contador] == 'E') es_seguro = true;
		contador++;
	}

	return es_seguro;
}



bool esta_segura(char lorien[MAX_ALTO][MAX_ANCHO], int alto, int ancho){

	bool segura = true; // asumo que es segura

	for (int i = 0; i < ancho; i++){
		for (int j = 0; j < alto; j++){
			if (!casillero_seguro(lorien, i, j))
				segura = false;
		}
	}

	return segura;
}
```

2)
a)
Aragorn le dará la hectarea a quien participo en ambas y mato como minimo a 10 orcos o 2 uruk.
Una interseccion...

```
void merecedores_de_la_hectarea(guerrero_t pelennor[MAX_GUERREROS], int
tope_p, guerrero_t cuernavilla[MAX_GUERREROS], int tope_c, guerrero_t
merecedores[MAX_GUERREROS], int* tope_m){
	(*tope_m) = 0;

	int i = 0, j = 0;
	while(i < tope_p && j < tope_m){
		if (strcmp(pelennor[i].nombre, cuernavilla[j].nombre) < 0){
			i++;
		} else if (strcmp(pelennor[i].nombre, cuernavilla[j].nombre)
== 0){
			if((pelennor[i].orcos_matados +
cuernavilla[j].orcos_matados >= 10) || pelennor[i].uruk_matados +
cuernavilla[j].uruk_matados >= 2){
				merecedores[(*tope_m)] = pelennor[i];
				merecedores[(*tope_m)].orcos_matados +=
cuernavilla[j].orcos_matados;
				merecedores[(*tope_m)].uruk_matados +=
cuernavilla[j].uruk_matados;
				(*tope_m)++;
			}
			i++;
			j++;
		} else {
			j++;
		}
	}
}
b)
En la union:
- El maximo es la suma de los topes, se da cuando los vectores son
distintos.
- El minimo es el mayor de los topes, uno contiene al otro.
En la diferencia simetrica:
- El maximo es la suma de los topes, se da cuando los vectores son
distintos.
- El minimo es 0, se da cuando son iguales.

3)
a)
Pide ordenar segun una velocidad que debemos calcular.
Podemos crear una funcion que calcule dicha velocidad para clarificar el
codigo.

float velocidad_enano(float velocidad_inicial, elemento_t
mochila[MAX_ELEMENTOS], int tope_m){
	float velocidad_elementos = 0;
	for (int i = 0; i < tope_m; i++){
		velocidad_elementos += mochila[i].peso;
	}

	return velocidad_inicial + velocidad_elementos / tope_m;
}

void ordenar_manada(integrante_t enanos[MAX_INTEGRANTES], int tope_e){

	for (int i = 0; i < tope_e; i++){
		for(int j = 0; j < tope_e - i - 1; j++){
			if (velocidad_enano(enanos[j]) >
velocidad_enano(enanos[j+1])){
				integrante_t enano_auxiliar = enanos[j];
				enanos[j] = enanos[j+1];
				enanos[j+1] = enano_auxiliar;
			}
		}
	}
}
```

b)
Tenemos que agregar un enano ordenadamente, reutilizamos la funcion que calcula la velocidad.
Empezamos desde el fondo corriendo una posicion hasta que encontramos donde va.

```
void ingresar_enano(integrante_t enanos[MAX_INTEGRANTES], int tope_e,
integrante_t enano_nuevo){
        int i = tope_e - 1;
        while(velocidad_enano(enano_nuevo) < velocidad_enano(enanos[i])){
                enanos[i+1] = enanos[i];
                i--;
        }

        enanos[i+1] = enano_nuevo;
}
```

4)
a)
La funcion determina si estan ordenados ascendentemente, o sea de menos a mas cerveza en el vaso.

```
bool estan_ordandas(float vasos[MAX_VASOS], int tope_v, int actual){
        if (actual >= tope_v-1) return true;

        if (vasos[actual] > vasos[actual+1]) return false;

        return estan_ordandas(vasos, tope_v, actual+1);
}
```

b)
La condicion de corte, en este caso tiene 2, que llegue al final del vector o que estan desordenados.
Si llegue al final del vector es porque estan ordenados.
Si estan desordenados, no tendria que seguir.

La llamada recursiva, es la 3er linea dentro de la rutina, hace el llamado con el proximo vaso.

5)
```
typedef struct coordenada{
        float latitud;
        float longitud;
} coordenada_t;

typedef struct arbol{
        coordenada_t posicion;
        int edad;
} arbol_t;

typedef struct ent{
        int numero_documento;
        char nombre[MAX_NOMBRE];
        arbol_t arboles_que_cuida[MAX_ARBOLES];
        int cantidad_arboles;
        float altura;
        coordenada_t posicion_actual;
        bool esta_en_movimiento;
        char rango[MAX_RANGO];
        bool tiene_voz_gruesa;
        bool da_frutos_ricos;
        int veces_que_lo_meo_un_perro;
} ent_t;
```