UTN - FRBA
Departamento De Electrónica

FINAL INFORMATICA I	***Duración***	***Fecha***	***Hojas***

Nombre y Apellido	***Nº Legajo***	***Calificación***		***Docente Evaluador***	
		número	***letras***	***Nombre***	***Firma***

Numere las hojas entregadas, y complete el casillero ***Hojas*** *con la cantidad.*
Lea detenidamente el enunciado, su correcta interpretación forma parte de esta evaluación.

Sección teórica

1. ¿Cual es el riesgo de la recursividad?
    a. El programa se vuelve más rápido de lo esperado
    b. El compilador entra en un bucle infinito
    c. Se puede producir un desbordamiento de pila
    d. Se libera la memoria dinámica
    e. Ninguna de las anteriores

2. ¿Qué resultado tiene esta expresión lógica: (5 > 3) && (2 < 1)?
    a. true
    b. false
    c. 1
    d. 0
    e. Ninguna de las anteriores

3. Indique que imprime el siguiente código si ingreso el numero 4

```
#include <stdio.h>

int main()
{
float numero;
int *p = (int*)&numero;

    printf("Ingrese un número: ");
    scanf("%d", p);

    if (numero % 2 == 0) {
        printf("El número %d es par.\n", numero);
    } else {
        printf("El número %d es impar.\n", numero);
    }

    return 0;
}
```

    a. El número 4 es par .
    b. El número 4 es impar.
    c. Genera una violacion de segmento.
    d. No compila.
    e. Ninguna de las anteriores

Sección práctica

1. Implemente una función que reciba un puntero a un vector de unsigned char y realice la conversión de cada elemento a un vector de unsigned short generado dinámicamente según la siguiente tabla. Los bits de la entrada marcados como S, A, B, C y D deben copiarse a la salida en la posición indicada en la tabla.

Entrada								Salida																
S	0	0	0	A	B	C	D	S	0	0	0	0	0	0	0	1	A	B	C	D	0	0	0
S	0	0	1	A	B	C	D	S	0	0	0	0	0	0	1	A	B	C	D	0	0	0	0
S	0	1	0	A	B	C	D	S	0	0	0	0	0	1	A	B	C	D	0	0	0	0	0
S	0	1	1	A	B	C	D	S	0	0	0	0	1	A	B	C	D	0	0	0	0	0	0
S	1	0	0	A	B	C	D	S	0	0	0	1	A	B	C	D	0	0	0	0	0	0	0
S	1	0	1	A	B	C	D	S	0	0	1	A	B	C	D	0	0	0	0	0	0	0	0
S	1	1	0	A	B	C	D	S	0	1	A	B	C	D	0	0	0	0	0	0	0	0	0
S	1	1	1	A	B	C	D	S	1	A	B	C	D	0	0	0	0	0	0	0	0	0	0

El prototipo de la función es el siguiente:

```
unsigned short * companding(unsigned char *p, unsigned int cant)
```

Donde:

- p: puntero al vector a convertir
- cant cantidad de datos a convertir

Devuelve un puntero a la zona de memoria donde está el vector con el resultado de la conversión..

2. Implementar una función que reciba el nombre de un archivo de texto como parámetro, lea su contenido cuente la cantidad de caracteres consecutivos iguales que hay para luego colocar esa información en una estructura como la siguiente

```
struct rle_S {
    char caracter;
    int cantidad;
};
```

La función retorna un puntero a un vector de estructuras conteniendo toda la información procesada del archivo. Ejemplo: Si el archivo contiene aaabbbbccdaa el vector de estructuras contendrá la siguiente información:

Carácter	cantidad
a	3
b	4
c	2
d	1
a	2

Prototipo de la función

`struct rle_S`* rle(char *archivo, int *cant)

Donde:

archivo: Nombre del archivo

cant: Puntero a una variable tipo int en la cual la función debe escribir la cantidad de estructuras retornadas

Devuelve un puntero a un vector de estructuras con la información del archivo procesada. NULL si no pudo procesar el archivo.