Universidad Nacional de Ingeniería
Facultad de Ciencias
Escuela Profesional de Ciencia de la Computación
Introducción a la Programación Universidad

Cuarta Práctica Calificada

CC102-AB **Ciclo: 2017-1**

Normas:

1. El alumno entregará esta hoja de examen debidamente llenada con sus datos.
2. La solución de la prueba se guardarán en **Escritorio**, carpeta: **ApellidoNombreCodigo** (sin espacios en blanco), la pregunta **n** se guardará en el archivo: **n.c** (n = 1, 2, ..).
3. No se permite: El uso de celulares, internet, USB, ingresar después de 15 min. de iniciado el examen ni salir antes de la hora de finalización.
4. Todo acto anti-ético será amonestado y registrado en el historial del estudiante.

Apellidos : _____ Nombres : _____
Sección : ____ Grupo: ___

Sugerencia: Para leer una cadena de caracteres **cc** del teclado puede usar gets(cc); y la saldrá un **warning**; déjelo pasar por el exámen; pero si tiene tiempo de ser perfecto, puede usar leerString(**cc**, **n**):

Sintaxis	**Ejemplo**
Int leerString(char **cc**[ ], int **n**); **Entradas**: **cc**: un arreglo de caracteres n = longitud del arreglo **Proceso:** Lee el buffer hasta encontrar <enter> ó se completen n-1 caracteres, agrega '\0' al final de la cadena y desecha la cola del buffer. **Salida**: número da caracteres leídos	char arr[6]; leerString(arr, 6); // 6 es el tamaño del arreglo Si tipea: **a b**<enter> arr ← "**a b\0**" y limpia el buffer. Si tipea: **ab cde**<enter> arr contiene "**ab cd\0**" y limpia el buffer.. Si tipea: **ab cdefg**<enter> arr contiene "**ab cd\0**" y limpia el buffer.

```
int leerString(char cc[ ], int n){
   int c, m=0;
   while((c=getchar())!=10) if(m<n-1)cc[m++]= c;
   cc[m] = '\0';
   return m;
}
```

1. [5 ptos.] Un estudio de colas de pasajeros del Metropolitano requiere programar lo siguiente: En un intervalo de tiempo, a una estación, llegan grupos de n1, n2, … pasajeros, que se leen desde el teclado, terminar ingresando 0 pasajeros; n1, n2, … se guardan en memoria dinámica apuntada por *personas la cual crece 1 unidad por cada grupo que llega. Lea los datos, y los va grabando en *personas, reporte los datos, ordene ascendentemente y vuelva a reportarlos.
   Sugerencia: Puede utilizar la siguiente main() o algo similar:

```
void leer    (int **vPersonas, int *n);
void ordenar (int *vPersonas,  int  n);
void reportar(int *vPersonas,  int  n);
void main(void){
    int *vPersonas = NULL, n = 0;        // n = tamaño de la memoria dinámica
    leer    (&vPersonas, &n);
    reportar( vPersonas,  n);
    ordenar ( vPersonas,  n);
    printf ("Datos ordenados\n");
    reportar( vPersonas,  n);
    free(vPersonas);
}
```

2. [5 ptos.] Continuando con el estudio del Metropolitano (problema 1, si desea copie el programa 1.c y lo continua). Aloje 10 datos ordenados: 2, 2, 4, 4, 5, 5, 5, 6, 6, 7 en memoria dinámica apuntada por *personas. Se requiere la siguiente estadística de *personas:
   - Número de grupos que llegaron = 10
   - La moda (número que más se repite) y la cantidad de veces que se repite = 5, 3
   - El promedio de personas en todo el experimento = 4.6

   Sugerencia: Puede utilizar la siguiente main() o algo similar:

```
void inicio  (int **vPersonas, int n);
void reportar(int  *vPersonas, int n);
void main(void){
    int *vPersonas = NULL, n = 10;
    inicio  (&vPersonas, n);
    reportar( vPersonas, n);
    free(vPersonas);
}
```

3. [5 ptos.] El dueño del gimnasio “LYON GYM” desea automatizar la información de los deportistas que asisten a su gimnasio para realizar ejercicios. Los datos con que cuenta de cada deportista son: nombre, edad, peso (kg), altura. Escriba un programa que registre un nuevo cliente del gimnasio e imprima sus datos

4. [5 ptos.] Dada una estructura llamada complejo{ int real, int imaginario}. Realice un programa que sume dos números complejos, ingresados desde el teclado.

Cuarta Práctica Calificada

CC102-CD **Ciclo: 2017-1**

Sugerencia: Para leer una cadena de caracteres **cc** del teclado puede usar gets(cc); y la saldrá un **warning**; déjelo pasar por el exámen; pero si tiene tiempo de ser perfecto, puede usar leerString(**cc**, **n**):

Sintaxis	**Ejemplo**
Int leerString(char **cc**[ ], int **n**); **Entradas**: **cc**: un arreglo de caracteres n = longitud del arreglo **Proceso:** Lee el buffer hasta encontrar <enter> ó se completen n-1 caracteres, agrega ‘\0’ al final de la cadena y desecha la cola del buffer. **Salida**: número da caracteres leídos	char arr[6]; leerString(arr, 6); // 6 es el tamaño del arreglo Si tipea: **a b**<enter> arr ← "**a b\0"** y limpia el buffer. Si tipea: **ab cde**<enter> arr contiene "**ab cd\0"** y limpia el buffer.. Si tipea: **ab cdefg**<enter> arr contiene "**ab cd\0"** y limpia el buffer.

```
int leerString(char cc[ ], int n){
   int c, m=0;
   while((c=getchar())!=10) if(m<n-1)cc[m++]= c;
   cc[m] = '\0';
   return m;
}
```

1. [5 ptos.] Escriba un programa que lea un número entero **m** en base 10, para convertirlo a base binaria, guardando cada cifra binaria en un vector dinámico llamado *vBinario. El algoritmo es el siguiente:
   a. Ingresar el número entero **m** >= 0.
   b. A medida que se hace las divisiones sucesivas entre 2, se aumenta en 1 unidad el tamaño de la memoria dinámica apuntada por *vBinario y se guarda el residuo al final de *vBinario. Se recalcula m dividiéndolo entre 2.

c. El proceso termina cuando **m** = 0.
d. Se imprimen todos los valores de *vBinario de atrás para adelante y se verá el número binario.

2. [5 ptos.] Escriba un programa que lea dos 2 enteros positivos **m** y **n**, defina un área dinámica de tamaño m*n para alojar datos de una matriz **mm[m][n]** con valores **mm[i][j] = i+j**, finalmente imprima la matriz; ejemplo si m = 3, **n = 2**, **mm** será:

```
0 1
1 2
2 3
```

3. [5 ptos.] Escriba un programa que muestre iterativamente el siguiente menú:

¿Desea ingresar una población?
1.- Sí.
2.- No.
Seleccione una de las opciones:

Si se selecciona la opción 1, se pide ingresar el nombre de un país y su población (en unidades de millones). Luego, se muestran todos los datos ingresados hasta ese momento. Finalmente, si se selecciona la opción 2, el programa debe terminar.

Sugerencia: Puede utilizar la siguiente main() o algo similar:

```
struct pais {
    char nombre[25];
    int poblacion;
};

void ingresar_pais(struct pais **lista, int *N);
void mostrar_pais (struct pais  *lista, int  N);
int main() {
    struct pais *lista = NULL;
    int opc, N = 0;
    do {
        printf("\n ¿Desea ingresar la población de un pais?\n 1.- Sí.\n 2.- No.\n Seleccione una de las opciones: ");
        scanf("%d", &opc);
        switch(opc)    {
            case(1): ingresar_pais(&lista, &N);                    break;
            case(2): printf ("\n Chau.\n\n");       free (lista);  break;
            default: printf ("\n Ingreso incorrecto.\n\n");
        }
    } while(opc != 2);
}
```

4. [5 ptos.]E scriba un programa que lea, desde el teclado, los datos de una variable *struct* **Estudiante** {nombre, codigo, nota[5]} calcule e imprima el promedio de las 5 notas de un estudiante.