UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS
DEPARTAMENTO DE ESTADÍSTICA
Profesor: Luis Gómez
Noviembre 2022

Computación Estadística III
(220257)

1. Escriba un programa que muestre lo siguiente en pantalla.

   a) En C, la capitalización de las letras es importante.

   b) main es donde comienza la ejecución del programa.

   c) Abrir y cerrar paréntesis de llaves encierran las instrucciones de un programa.

   d) Todas las instrucciones deben terminar con un ";".

2. ¿Que salida esperaría del siguiente programa? Compruébelo.

```c
#include <stdio.h>
int main (void)
{
   printf ("Testing...");
   printf ("....1");
   printf ("...2");
   printf ("..3");
   printf ("\n");
   return 0;
}
```

3. Escriba un programa que reste 15 a 85 y muestre el resultado en pantalla con un mensaje apropiado.

4. Identifique los errores del siguiente programa. Corrijalos y ejecute el programa corregido.

```c
#include <stdio.h>
int main (Void)
(
   INT sum;
   /* COMPUTE RESULT
   sum = 25 + 37 - 19
   /* DISPLAY RESULTS //
   printf ("The answer is %i\n" sum);
   return 0;
}
```

5. ¿Qué salida obtendrá del siguiente programa? Compruébelo.

```c
#include <stdio.h>
int main (void)
{
   int answer, result;
   answer = 100;
   result = answer - 10;
   printf ("The result is %i\n", result + 5);
   return 0;
}
```

6. ¿Cuáles de los siguientes son inválidos como nombres de variable? ¿Por qué?

```
Int
Calloc
floating
ReInitialize
char
Xx
_1312
_
6_05
alpha_beta_routine
z
A$
```

7. Escriba un programa en C que convierta 27°F a Celsius usando la fórmula $C = (F - 32)/1.8$ y muestre el resultado.

8. ¿Qué salida esperaría del siguiente programa? Compruébelo.

```
#include <stdio.h>
int main (void)
{
   char c, d;

   c = 'd';
   d = c;
   printf ("d = %c\n", d);
   return 0;
}
```

9. Escriba un programa que evalúe el polinomio $3x^4 - 4x^2 + 3$ para x=3.22, y muestre el resultado.

10. Cree un programa que evalúe la siguiente expresión y muestre el resultado usando formato exponencial: $(3.31 \cdot 10^{-8} \cdot 2.01 \cdot 10^{-7})/(7.16 \cdot 10^{-6} + 2.01 \cdot 10^{-8})$

11. Para redondear un entero i al "siguiente múltiplo par mas grande"de otro entero j, se usa la formula next_multiple=i + j - i.

    Escriba un programa que encuentre el siguiente múltiplo mas grande, considerando:

i	j
365	7
12258	23
996	4

12. Escriba un programa C que reciba 3 valores A, B y C para calcular e imprimir el resultado de la expresión 3A+5B-7C.

13. Escriba un programa C que reciba 10 números e imprima solamente los positivos.

14. Escriba un programa que solicite al usuario el ingreso de los pesos, en kilogramos, de 5 ocupantes de un vehículo y emita un mensaje indicando el peso total y si tal peso total excede un cuarto de tonelada.

15. Escriba un programa que toma un número entero como parámetro, y devuelve 1 si es negativo o 0 si no lo es.

16. Escriba un programa que reciba el nombre, la edad, el sexo, el estado civil de cualquier persona e imprima el nombre de la persona, si corresponde a un hombre soltero, mayor de 30 años o a una mujer viuda menor de 50 años.

17. Escriba un programa que calcule el área y la circunferencia del círculo en función de valor del radio proporcionado por el usuario.

18. Escriba un programa que imprima el salario total de un obrero sabiendo el número de horas que trabajó en la semana, cuánto se le paga por cada hora y que se le hace una bonificación semanal en el salario de $25 por cada producto terminado con cero errores.

19. Escriba un programa que permita el ingreso de 15 números y que cuente cuántos de los números ingresados son pares.

20. Escriba un programa que permita el ingreso de 10 números y que muestre el mayor, el menor y el promedio.

21. Escriba un programa que dada la medida de un ángulo expresada en radianes, permita obtener su equivalente en grados, minutos y segundos. Por ejemplo, si la entrada de su función fuera 1, ella debe imprimir en pantalla:

    1 radian equivale a 57 grados, 17 minutos y 44 segundos.
    Nota: 2π radianes = 360 grados. 1 grado = 60 minutos. 1 minuto = 60 segundos.

22. Crear un programa C que reciba un entero n y resuelva lo siguiente:

$$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + ... + n$$

23. Escriba un programa que genere la serie $1, 5, 3, 7, 5, 9, 7, 11, 9, 13...$.

24. Crear un programa C que imprima la tabla de multiplicar de cualquier numero.

25. Crear un programa C que calcule el factorial de un número.

26. Crear un programa C donde el usuario ingrese 2 números y pueda ejecutar cualquiera de las operaciones básicas. Si se desea salir se debe ingresar 0.

27. Crear un programa C para calcular la media ponderada

28. Construya un programa que lea un símbolo "*", "#", "+", "-","&" y que despliegue cinco veces el símbolo ingresado. Si el símbolo ingresado no es ninguno de los enumerados se debe desplegar el mensaje "Símbolo no reconocido".
    Por ejemplo, si al ejecutar el programa se ingresa el símbolo "+" se debe desplegar:

    + + + + +

    Para producir la salida NO se puede usar comandos for ni while.

29. Construya un programa que lea un número entero y que despliegue como resultado un entero correspondiente al número ingresado con los dígitos invertidos. Por ejemplo, si al ejecutar el programa se ingresa el valor 204354309 el programa debe desplegar:

    903453402

30. Modifique el programa del problema anterior, para que funcione con enteros negativos. Si recibe -123, debe desplegar 123-

31. Escriba un programa que muestre por pantalla las raíces de la ecuación donde $ax^2 + bx + c = 0$, donde a,b y c son ingresados por el usuario. Considere todos los casos posibles.

32. Escriba un programa que lea un número entero N y que dibuje un cuadrado hueco de lado N asteriscos. Por ejemplo, si se ingresa 8 el programa debe dibujar:

33. Crear un programa donde el usuario pueda ingresar dos valores y opta por sumar, restar, dividir ó multiplicar dichos valores, utilizando switch-case.

34. Crear un programa C que pida al usuario un número de día y un número de mes, evalúe si la entrada es correcta y en caso de que así sea calcule el día del año que es, considerando un año no bisiesto. Por ejemplo, el 1 de febrero sería el día 32.

35. Crear un programa C que pida al usuario un número de día, un número de mes y año, evalúe si la entrada es correcta y en caso de que así sea calcule el día del año que es (NO olvide considerar los casos con años bisiesto).

36. Escriba un programa que imprima en pantalla una tabla ALINEADA de n vs n^2, para $n = 1, ..., 10$.

37. La suma de los primeros n enteros también se puede calcular usando la fórmula:

$$suma = n(n+1)/2$$

    Escriba un programa que calcule la suma cada 5 unidades desde 5 a 50 usando esta fórmula y despliegue el resultado en una pantalla.

38. El factorial de un número (no nulo) se calcula:

$$n! = 1 \times 2 \times 3 \times \cdots \times n$$

    Escriba un programa que calcule los primeros 10 factoriales, y los muestre en una tabla.

39. Reescriba el siguiente programa de forma mas legible, compile y ejecute:

```c
#include <stdio.h>
int main(void){
int n,two_to_the_n;
printf("TABLA DE POTENCIAS DE 2\n\n");
printf(" n       2^n\n\n");
printf("---      ---------------\n");
two_to_the_n=1;
for(n=0;n<=10;n++){
printf("%2i      %i\n",n,two_to_the_n); two_to_the_n*=2;}
return 0;}
```

40. Un punto decimal antes de la especificación de ancho en prinf (%.2i) tiene un propósito especial. Encuentre el propósito al ejecutar el siguiente programa:

```c
#include <stdio.h>
int main (void)
{
   int dollars, cents, count;
   for ( count = 1; count <= 10; ++count ) {
      printf ("Enter dollars: ");
      scanf ("%i", &dollars);
      printf ("Enter cents: ");
      scanf ("%i", &cents);
      printf ("$%i.%.2i\n\n", dollars, cents);
   }
   return 0;
}
```

41. Escriba un programa que entregue la suma de los dígitos de un entero proporcionado por el usuario a través de la terminal.

42. Cree un programa que reciba dos enteros desde la terminal y decida si el primero es divisible por el segundo, entregando el mensaje apropiado.

43. Cree un programa que reciba dos enteros desde la terminal, y entregue el resultado de la división aproximado a tres valores decimales. Debe chequear división por 0.

44. Cree un programa que reciba un entero no negativo, y entregue los dígitos en palabras. Por ejemplo, si la entrada es 123, debe desplegar: uno dos tres.

45. Escriba un programa que actúe como una "calculadora de papel". El programa debe recibir dos entradas de la forma

    numero operador

    Las siguientes operaciones deben ser reconocidas por el programa:

    + - ∗ / S E

    El operador S indica que se debe iniciar el "acumulador".

    El operador E termina la ejecución del programa.

    Las operaciones aritméticas son realizadas en el contenido del acumulador versus el número que se está ingresando. A continuación un ejemplo de ejecución en la terminal:

Inicio de cálculos		
10 S	Inicia el acumulador a 10	
$=$ 10.000000	Contenido del acumulador	
2 /	Divide por 2	
$=$ 5.000000	Contenido del acumulador	
55 $-$	Resta 55 $=$ -50	Contenido del acumulador
100.25 S	Inicial el acumulador a 100.25	
$=$ 100.25	Contenido del acumulador	
4 $*$	Multiplica por 4	
$=$ 401.0000	Contenido del acumulador	
0 E	Fin del programa	
Fin de cálculos		

46. Cree un programa C que verifique si un entero ingresado por el usuario es un palíndromo (al invertir dígitos resulta ser el mismo número).

47. Cree un programa C que evalúe la siguiente función por tramos:

$$f(x) = \begin{cases} x^3 & \text{, si } x < -2 \\ x^2 & \text{, si } x \geq -2 \text{ y } x < 3 \\ x & \text{, si } x \geq 3 \end{cases}$$

48. Escriba un programa que calcule la media de un arreglo de 10 valores punto flotante.

49. La Criba de Eratóstenes permite encontrar todos los números primos menores o iguales que un entero n. El algoritmo es el siguiente:

    a) Defina un array entero P de tamaño n, inicialiciado a 0.

    b) Inicie i a 2.

    c) Si $i > n$, el algoritmo termina.

    d) Si $P[i]$ es 0, entonces i es primo.

    e) Para todos los enteros j tal que $i \times j = n$, asignar $P[i \times j]$ a 1.

    f) Sume 1 a i y vuelva al paso c.

50. ¿Qué salida espera del siguiente programa?

```c
int main (void)
{
  int numbers[10] = { 1, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  int i, j;

  for ( j = 0; j < 10; j++ ){
    for ( i = 0; i < j; ++i )
      numbers[j] += numbers[i];
  }

  for ( j = 0; j < 10; ++j )
    printf ("%i ", numbers[j]);

  printf ("\n");
  return 0;
}
```

51. Escriba un programa en C para leer dos valores en un arreglo (array) unidimensional y mostrarlo en orden inverso. Ejemplo: Ingresamos los elementos 2 y 5 y la salida esperada : 5 y 2

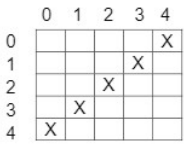
52. Escriba un programa en C para imprimir todos los elementos negativos de un arreglo unidimensional .

53. Escriba un programa en C para encontrar la suma de todos los elementos de una arreglo unidimensional.

54. Rellene un array con los 100 primeros números enteros y los muestre en pantalla en orden ascendente.

55. Escriba un programa en C que lea 10 números por teclado, los almacene en un array y muestre la media.

56. Escriba un programa en C donde ingreses un nombre y muestre la longitud del nombre ingresado.



57. Escriba un programa en C que cuente cuantas vocales posee un string.

58. Escriba un programa en C que invierta un nombre.

59. Escriba un programa en C para ingresar elementos en un arreglo unidimensional por el usuario y cuente los elementos pares e impares.

60. Escriba un programa en C para ingresar elementos en un arreglo unidimensional y cuente los elementos negativos.

61. Escriba un programa en C para ingresar elementos en un arreglo unidimensional e imprima todos los elementos únicos .

62. Escriba un programa en C para ingresar elementos en un arreglo unidimensional por usuario y cuente elementos duplicados.

63. Escriba un programa en C que solicite al usuario el ingreso de dos vectores a y b, y luego muestre por pantalla el producto vectorial axb. El programa debe mostrar además un mensaje indicando si los vectores son paralelos o no.

64. Crear un programa donde se introduzca 2 matrices de 4x4 y que entregue el resultado de la suma de ambas matrices.

65. Ingresar datos en una matriz de enteros de 5x5 e informar los N indicados con una X.

66. Escriba un programa en C que cree un array bidimensional de longitud 5x5 , de forma que los componentes pertenecientes a la diagonal de la matriz tomen valor uno y el resto cero. Muestre el contenido del array en pantalla.

67. Crear un programa en C que cree un array bidimensional de longitud 5x5 inicialice el array de tal forma que el valor de cada elemento sea la suma del número de su fila y del número de su columna, Muestre el contenido del array en pantalla.

68. Escriba un programa en C para leer elementos de una matriz y realice una multiplicación escalar de la matriz.

69. Escriba un programa en C para leer elementos en dos matrices y multiplíquelos.

70. Escriba un programa en C para leer elementos en una matriz y verifique si la matriz es una matriz triangular superior o no.

71. Escriba un programa en C para leer elementos en una matriz y verifique si la matriz es una matriz triangular inferior o no.

72. Escriba un programa en C que sea capaz de intercambiar 2 filas ó dos columnas de una matriz de 4x4.-

73. Escriba un programa en C que calcule y muestre por pantalla, en formato matricial, la suma 3A + 2B, donde A y B son matrices cuadradas de orden 6 ingresadas por el usuario. El programa además debe mostrar por pantalla la traza de 3A + 2B.

74. Modifique el programa sqrt_NR_3.c visto en clases, para que ahora el parámetro de precisión e sea preguntado al usuario. Experimente con distintos valores de e para ver el resultado en el cálculo de la raíz.

75. Modifique el programa anterior para que en cada iteración, se imprima el valor de *aprox* y se visualice la convergencia.

76. Escriba una función recursiva que eleve un entero a una potencia entera. Defina el retorno como long int e inclúyala en el programa main. Los datos necesarios debe entregarlos el usuario a través de la terminal.

77. Escriba un programa con el adecuado uso de funciones que calcule el mínimo común múltiplo entre dos enteros. Recuerde que: $mcm(u, v) = u * v/mcd(u, v)$. Donde mcd es el máximo común divisor entre los 2 enteros.

78. Escriba un programa con el adecuado uso de funciones que determine si un numero es primo o no.

79. Escriba una función que reciba como argumento una matriz de tamaño 4×5 y una de tamaño 5×4, y almacene la transpuesta de la primera matriz en la segunda.

80. Modifique la función anterior para usar arreglos de tamaño variable. Ahora la función debe tener como argumentos el numero de filas y columnas.

81. Realizar una función llamada par, que toma un número entero como parámetro, y devuelve 1 si es par o devuelve 0 si es impar. Observación: Para saber si un número entero es par, al dividirlo entre 2 su resto debe ser 0.

82. Que permita leer el valor correspondiente a una distancia en kilómetros y las visualice expresadas en metros

83. Escribir una función que reciba por valor los catetos de un triángulo rectángulo y devuelva la hipotenusa del mismo.

84. Realizar una función llamada media, que toma dos números reales como parámetros, y devuelve un número real que es la media de los dos números pasados como parámetros.

85. Realizar una función llamada ultima, que toma una cadena de hasta 10 caracteres como parámetro, y devuelve el último carácter.

86. Escriba un programa en C para ingresar cualquier número del usuario y encuentre el cubo del número dado usando la función.

87. Escriba un programa en C para ingresar el radio del círculo por el usuario y encuentre el diámetro y el área del círculo dado usando la función.

88. Calcule el área y el perímetro de un rectángulo dada la base y la altura.

89. A un trabajador le pagan según sus horas trabajadas y la tarifa está a un valor por hora. Si la cantidad de horas trabajadas es mayor a 40 horas, la tarifa por hora se incrementa en un 50 %para las horas extras. Calcular el salario del trabajador dadas las horas trabajadas y la tarifa.

90. Escriba un programa en C para almacenar e imprimir el curso, nombre, edad y nota de un estudiante usando estructuras.

91. Escriba un programa en C que lea del teclado los datos de una variable tipo struct llamada estudiante , calcule e imprime el promedio de las 5 notas de un estudiante.

92. Escriba un programa en C para sumar, restar y multiplicar dos números complejos usando estructuras

93. Escriba un programa en C que lea en un array de estructuras los datos de los N trabajadores de una empresa y que imprima los datos del empleado con mayor y menor sueldo.

94. Escriba un programa en C para comparar dos fechas ingresadas por el usuario. Haga una estructura llamada Fecha para almacenar los elementos día, mes y año para almacenar las fechas. Si las fechas son iguales, muestre "Las fechas son iguales", de lo contrario, visualice "Las fechas no son iguales".

95. Cree un programa C que calcule el numero de días entre dos fechas.

    Indicaciones:

    - El numero de días entre dos fechas, $f1$ y $f2$, se calcula restando los valores de N respectivos de acuerdo a la formula:

    $$N = 1461f(year, month)/4 + 153g(month)/5 + day$$

    - $f(year, month) = year - 1$, si $month \leq 2$ y $f(year, month) = year$, en otro caso.
    - $g(month) = month + 13$, si $month \leq 2$ y $g(month) = month + 1$, en otro caso.
    - La formula anterior se utiliza para fechas posteriores al 1 de marzo de 1900.
    - Estructure el programa lógicamente en funciones separadas. Utilice la estructura fecha definida en clases. Entregue un mensaje de error en caso de que alguna de las fechas sea anterior al 1 de marzo de 1700.

96. Cree un programa que realice lo anterior, pero para fechas posteriores al 1 de marzo de 1700, considerando que:

    - Para fechas entre el marzo 1 de 1800 a 28 de febrero de 1900 se debe sumar 1 a N.
    - Para fechas entre el marzo 1 de 1700 a 28 de febrero de 1800 se debe sumar 2 a N.

97. Cree un programa que reciba 2 estructuras de tiempo (hh:mm:ss) y calcule el tiempo (hh:mm:ss) que ha transcurrido entre ellas. Cuidado con los tiempos que pasan la medianoche. Por ejemplo, el tiempo transcurrido entre las 3:45:15 y las 9:44:03 es 5:58:48.

98. Cree un programa que reciba una estructura dateAndTime, ejecute la función timeUpdate. Si la hora llega a medianoche, entonces debe ejecutar la función dateUpdate. Debe mostrar la estructura dateAndTime resultante.

99. Cree una función que se llame *substring*, para extraer una porción de una cadena de texto. Se debe llamar de la forma:

    ```
    substring(source, start, count, result);
    ```

    donde:

    - source: es la cadena de texto original.
    - start: es el indice desde el cual se comienza a extraer el substring
    - count: indica cuantos caracteres se deben extraer desde start
    - result: contiene el substring extraído.

100. Escriba una función llamada *findString* que determine si un substring existe dentro de otro. El primer argumento deber ser el string sobre el que se va a buscar y el segundo el string a buscar. Si lo encuentra, debe retornar la posición y en caso contrario un -1;

101. Escriba una función llamada *removeString*, que remueve una cantidad especificada de caracteres de un string. Debe recibir 3 argumentos, el string fuente, la posición desde donde se debe remover, y la cantidad de caracteres a remover.

102. Cree una función llamada *insertString* para insertar un string dentro de otro. Los argumentos deben el string sobre el que se debe insertar, la posición y el string a insertar.

103. Escriba una función llamada *replaceString* (utilizando las tres funciones creadas anteriormente), que tome 3 strings como argumentos: string, s1 y s2, de forma tal que reemplace s1 dentro de string por s2. Ademas, la función debe retornar si se pude hacer el reemplazo o no, para por ejemplo, reemplazar todas las instancias de un caracter de la forma:

```
do
  stillFound = replaceString (text, " ", "");// elimina espacios
while ( stillFound = true );
```

104. Escriba una función llamada *dictionarySort* que ordene alfabéticamente un array de estructuras *entry* (un diccionario) de las que se vio en clases.

105. Extienda la función *strToInt* cista en clases para que ahora funcione con enteros negativos (la cadena a convertir empieza con ").

106. Escriba una función llamada *strToFloat* que convierta un string en variable de punto flotante. Debe funcionar también para números negativos.

107. Si *c* es una letra en minúsculas, la expresión:

```
c - 'a' + "A"
```

produce el equivalente en mayúsculas.

Escriba una función llamada *uppercase* que convierta todos los caracteres de un string a mayúsculas.

108. Escriba una función llamada *intToStr*, que convierta un entero (incluido negativos) a una cadena de texto.

109. Escriba un programa en C que cuente la coincidencia de 1 carácter dentro de un string.

110. Escriba un programa en C que cuente cuántas vocales se encuentran dentro de un string.

111. Escriba un programa en C que invierta una string.

112. Escriba un programa en C que reemplace todos los espacios en blanco de una string por un asterisco.

113. Escribir un programa en C que cuente el número de palabras en un string.

114. Escribir un programa en C lea el nombre y los dos apellidos de una persona (en tres cadenas de caracteres diferentes) y unirlo.

115. Escribir un programa en C que convierta una cadena en minúsculas a mayúsculas.

116. Escribir un programa en C que invierta el orden de las palabras en una string.

117. Escribir un programa en C que encuentre el carácter máximo que ocurre en un string.

118. Escribir un programa en C que encuentre el carácter mínimo que ocurre en un string.

119. Escribir un programa en C que cuente la frecuencia de cada carácter en una cadena en una única cadena.

120. Escriba un programa en C que compare dos string y si son iguales retorne 1, en caso contrario que retorne 0.

121. Escribir un programa en C que calcule el sueldo que le corresponde a un trabajador de una empresa que cobra 40000 euros anuales, el programa debe realizar los cálculos en función de los siguientes criterios:

    - Si lleva más de 10 años en la empresa se le aplica un aumento del 10
    - Si lleva menos de 10 años pero más que 5 se le aplica un aumento del 7
    - Si lleva menos de 5 años pero más que 3 se le aplica un aumento del 5
    - Si lleva menos de 3 años se le aplica un aumento del 3

122. Escribir un programa que calcule el dígito verificador del RUT.

123. Escribir un programa que determine la intersección de dos conjuntos numéricos almacenados en dos arreglos.

124. Escriba una función que se llame insertEntry, para insertar una nueva entrada en una lista encadenada. La función debe recibir como entrada un puntero al elemento que va a ser insertado (de tipo struct entry definido en clases), y un puntero al elemento de la lista al cual va a ser insertado después de él. Verifique recorriendo la lista generada de que la inserción fue correcta.

125. La función desarrollada en el ejercicio anterior, solo inserta elementos después de un elemento existente en la lista, evitando insertar elementos al inicio de la misma. Utilice la misma función para insertar un elemento al inicio de la lista.

126. Escriba una función llamada removeEntry para remover un elemento de una lista encadenada. Debe funcionar también en el caso de querer remover el primer elemento de la lista. El único argumento de la función debe ser un puntero al elemento anterior a remover.

127. Una lista doblemente encadenada es un lista en la cual cada entrada contiene un puntero al elemento anterior y uno al posterior en la lista. Defina una estructura apropiada que implemente esto y una función que recorra e imprima los valores de la lista, hacia adelante y hacia atrás.

128. Desarrolle las funciones insertEntry y removeEntry de los problemas anteriores para una lista doblemente encadenada. Notar que ahora la función removeEntry puede recibir como entrada directamente la entrada a eliminar. Debe funcionar también en el caso del primer y ultimo elemento de la lista.

129. Considere la función sort utilizada en el siguiente programa:

```c
#include <stdio.h>
int main (void)
{
  void sort (int a[], int n);

  int i, dim;

  printf ("Cuantos valores va a ingresar?: \n");
  scanf("%i",&dim);

  int array[dim];

  for ( int i = 0; i < dim; i++ ){
    printf("valor #%i: ",i+1);
    scanf ("%i", &array[i]);
  }
```

```
17
18    sort (array, dim);
19
20    printf ("\narray ordenado:\n");
21    for ( i = 0; i < dim; i++ )
22      printf ("%i ", array[i]);
23
24    printf ("\n");
25    return 0;
26  }
27
28  void sort (int a[], int n)
29  {
30    int i, j, temp;
31
32    for ( i = 0; i < n - 1; i++ )
33      for ( j = i + 1; j < n; j++ )
34        if ( a[i] > a[j] ) {
35          temp = a[i];
36          a[i] = a[j];
37          a[j] = temp;
38        }
39  }
```

Desarrolle una versión de la función (y modifique el programa de forma correspondiente) que utilice punteros.

130. Escriba una función que ordene 3 enteros en forma ascendente sin utilizar arreglos. Desarrolle un programa que utilice la función desarrollada.

131. Considere la función readLine utilizada en el siguiente programa:

```
1  #include <stdio.h>
2  #include <stdbool.h>
3
4  int main (void)
5  {
6    int countWords (const char string[]);
7    void readLine (char buffer[]);
8
9    char text[81];
10   int totalWords = 0;
11   bool endOfText = false;
12
13   printf ("Ingrese su texto.\n");
14   printf ("Cuando termine, presione 'ENTER'.\n\n");
15
16   while ( ! endOfText ){
17     readLine (text);
18
19     if ( text[0] == '\0' )
20       endOfText = true;
21     else
22       totalWords += countWords (text);
23   }
24
25   printf ("\nHay %i palabras en el texto.\n",totalWords);
26   return 0;
27 }
28
```

```
29 int countWords (const char string[])
30 {
31   int i, wordCount = 0;
32   bool lookingForWord = true, alphabetic (const char c);
33
34   for ( i = 0; string[i] != '\0'; ++i ){
35     if ( alphabetic(string[i]) ){
36       if ( lookingForWord ){
37         wordCount++;
38         lookingForWord = false;
39       }
40     }
41     else
42       lookingForWord = true;
43   }
44
45   return wordCount;
46 }
47
48 bool alphabetic (const char c)
49 {
50   if ( (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') )
51     return true;
52   else
53     return false;
54 }
55
56 void readLine (char buffer[])
57 {
58   char character;
59   int i = 0;
60
61   do {
62     character = getchar ();
63     buffer[i] = character;
64     i++;
65   } while ( character != '\n' );
66
67   buffer[i - 1] = '\0';
68 }
```

Desarrolle una versión de la función (y modifique el programa de forma correspondiente) que utilice punteros a char en vez de un arreglo.

132. Considere la función compareStrings utilizada en el siguiente programa:

```
1 #include <stdio.h>
2
3 struct entry
4 {
5   char word[15];
6   char definition[150];
7 };
8
9 int main(void)
10 {
11   const struct entry dictionary[100] =
12     { {"adiós", "El concepto se emplea a modo de saludo, cuando una
          persona se despide de otra."},
```

```
13        {"bata", "Prenda de vestir holgada, con mangas y abierta por
              delante, que se usa al levantarse y para estar por casa"},
14        {"casa", "Edificación construida y destinada para ser habitada"},
15        {"dedo", "Los dedos son las porciones distales de la región de la
              mano y del pie del ser humano y de otros animales"},
16        {"torta", "Masa de harina y otros ingredientes que se cuece a
              fuego lento y que usualmente tiene forma redondeada"}
17      };
18
19    int lookup (const struct entry dictionary[], const char search[],
          const int entries);
20
21    char word[15];
22    int entry, entries = 5;
23
24    printf("Ingrese palabra: ");
25    scanf("%14s",word);
26
27    entry = lookup (dictionary, word, entries);
28
29    if (entry != -1){
30      printf("%s\n", dictionary[entry].definition);
31    } else {
32      printf("La palabra no existe en el diccionario\n");
33    }
34
35    return 0;
36  }
37
38  int lookup (const struct entry dictionary[], const char search[], const
        int entries)
39  {
40    int compareStrings (const char s1[], const char s2[]);
41
42    int low = 0, high = entries -1, mid, result;
43
44    while ( low <= high ){
45      mid = (low + high)/2;
46      result = compareStrings(dictionary[mid].word, search);
47
48      if ( result == -1 )
49        low = mid +1;
50      else if (result == 1)
51        high = mid - 1;
52      else
53        return mid;
54    }
55
56    return -1;
57  }
58
59  int compareStrings (const char s1[], const char s2[])
60  {
61    int i = 0, answer;
62
63    while ( s1[i] == s2 [i] && s1[i] != '\0' && s2[i] != '\0' )
64      i++;
65
66    if ( s1[i] < s2[i] )
```

```
    answer = -1;
  else if ( s1[i] == s2[i] )
    answer = 0;
  else
    answer = 1;

  return answer;
}
```

Desarrolle una versión de la función (y modifique el programa de forma correspondiente) que utilice punteros a char en vez de arreglos.

133. Considere la estructura de datos siguiente:

```
struct date
{
    int month;
    int day;
    int year;
}
```

Escriba una función llamada dateUpdate que tome un puntero a una estructura de este tipo como argumento y que actualice la fecha al día siguiente. No olvidar considerar los casos de fin de mes y fin de año.

134. Considere las siguientes declaraciones:

```
char *message = "Programming in C is fun\n";
char message2[] = "You said it\n";
char *format = "x = %i\n";
int x = 100;
```

Determine si cada llamada a la función printf en los siguientes conjuntos instrucciones produce la misma salida que el resto:

```
/*** set 1 ***/
printf ("Programming in C is fun\n");
printf ("%s", "Programming in C is fun\n");
printf ("%s", message);
printf (message);

/*** set 2 ***/
printf ("You said it\n");
printf ("%s", message2);
printf (message2);
printf ("%s", &message2[0]);

/*** set 3 ***/
printf ("said it\n");
printf (message2 + 4);
printf ("%s", message2 + 4);
printf ("%s", &message2[4]);

/*** set 4 ***/
printf ("x = %i\n", x);
printf (format, x);
```

135. Defina una macro llamada MIN que entregue el mínimo entre 2 valores. Escriba un programa para probar la macro.

136. Defina una macro llamada MAX3 que entregue el máximo entre 3 valores. Escriba un programa para probar la macro.

137. Defina una macro llamada IS_UPPER_CASE que entregue un valor 0 si un carácter esta en minúscula. Escriba un programa para probar la macro.

138. Defina una macro llamada IS_ALPHABETIC que entregue un valor no 0 si un carácter es alfabético. Utilice las macros IS_UPPER_CASE del problema anterior y IS_LOWER_CASE definida en clases. Escriba un programa para probar la macro.

139. Defina una macro llamada IS_DIGIT que entregue un valor no 0 si un carácter es un dígito del '0' al '9'. Escriba un programa para probar la macro.

140. Defina una macro llamada IS_SPECIAL que entregue un valor no 0 si un carácter es especial, es decir, no es alfabético ni dimito. Utilice las macros IS_ALPHABETIC y IS_DIGIT de los problemas anteriores. Escriba un programa para probar la macro.

141. Escriba una macro llamada ABSOLUTE_VALUE que calcule el valor absoluto de su argumento. Asegure que una expresión como:

```
1 ABSOLUTE_VALUE(x+delta)
```

    se evalúe apropiadamente.

142. Modifique los programas realizados anteriormente, para que puedan ser llamados desde R a través de la función .C.

143. Modifique los programas realizados anteriormente, para que puedan ser llamados desde R a través de la función .Call.

144. Modifique los programas realizados anteriormente, para que puedan ser llamados desde R a través de la función .External.

145. Para cada uno de los códigos R desarrollados anteriormente, implemente, de ser posible, alguna de las técnicas de paralelización vistas en clases.

146. Realice lo siguiente:

    a) Considere el dataset `penguins` del paquete `palmerpenguins`. Este contiene medidas de 3 especies de pinguinos adultos presentes en la Antártica. Utilice la técnica de random forests (paquete `ranger`, conjunto de árboles de desición) para ajustar un modelo de clasificación, con etiquetas la especie, y predictores las siguientes medidas: `bill_length_mm`, `bill_depth_mm`, `flipper_length_mm`, y `body_mass_g`. Utilice calculo paralelo para estimar el mejor modelo.

    b) Obtenga un intervalo de confianza para el score de importancia de cada predictor de los random forests generados con `ranger()`.