RELACIÓN DE PROBLEMAS I. Introducción a C++

1. Indique cuál sería el resultado de las siguientes operaciones:

```
int salario_base;
int salario_final;
int incremento;

salario_base = 1000;
salario_final = salario_base;

incremento = 200;
salario_final = salario_final + incremento;

salario_base = 3500;

cout << "\nSalario base: " << salario_base;
cout << "\nSalario final: " << salario_final;
```

   Responda razonadamente a la siguiente pregunta: ¿El hecho de realizar la asignación `salario_final = salario_base;` hace que ambas variables estén ligadas durante todo el programa y que cualquier modificación posterior de `salario_base` afecte a `salario_final`?

   Finalidad: Ejemplo básico de asignación a una variable del resultado de una expresión.. Dificultad Baja.

2. Crear un programa que pida un valor de intensidad y resistencia e imprima el voltaje correspondiente, según la *Ley de Ohm*:

   voltaje = intensidad * resistencia

   Finalidad: Asignar a una variable el resultado de una expresión. Dificultad Baja.

3. Cread un programa que nos pida la longitud del radio, calcule el área del círculo y la longitud de la circunferencia correspondientes, y nos muestre los resultados en pantalla. Recordad que:

   $$\text{longitud circunferencia} = 2\pi r \qquad \text{área circulo} = \pi r^2$$

   Usad el literal `3.1416` a lo largo del código, cuando se necesite multiplicar por π.

   Una vez hecho el programa, cambiad las apariciones de `3.1416` por `3.1415927`, recompilad y ejecutad.

   Finalidad: Conciernarnos del problema de los literales repetidos a lo largo del código. Dificultad Baja.

4. Un banco presenta la siguiente oferta. Si se deposita una cantidad de euros `capital` durante un año a plazo fijo, se dará un interés dado por la variable `interes`. Realizad un programa que lea una cantidad `capital` y un interés `interes` desde teclado y calcule en una variable `total` el dinero que se tendrá al cabo de un año, aplicando la fórmula:

$$\texttt{total} = \texttt{capital} + \texttt{capital} * \frac{\texttt{interes}}{100}$$

   Es importante destacar que el compilador primero evaluará la expresión de la parte derecha de la anterior asignación (usando el valor que tuviese la variable `capital`*) y a continuación ejecutará la asignación, escribiendo el valor resultante de la expresión dentro de la variable* `total`*).*

   A continuación, el programa debe imprimir en pantalla el valor de la variable `total`. Tanto el capital como el interés serán valores reales. Supondremos que el usuario introduce el interés como un valor real entre 0 y 100, es decir, un interés del $5{,}4\,\%$ se introducirá como `5.4`. También supondremos que lo introduce correctamente, es decir, que sólo introducirá valores entre 0 y 100.

   Supongamos que queremos modificar la variable original `capital` con el nuevo valor de `total`. ¿Es posible hacerlo directamente en la expresión de arriba?

   Finalidad: Resolver un problema real sencillo, usando varias sentencias. *Dificultad Baja.*

5. Construya un programa para leer el valor de una variable `salario_base` de tipo `double`, la incremente un 2 % e imprima el resultado en pantalla. Para realizar este cálculo, multiplique por `1.02` el valor original. Tiene varias alternativas:

   a) Calcular `1.02 * salario_base` dentro de la sentencia `cout`

   b) Introducir una variable `salario_final`, asignarle la expresión anterior y mostrar su contenido en la sentencia `cout`

   c) Modificar la variable original `salario_base` con el resultado de incrementarla un 2 %.

   Indique qué alternativa elige y justifíquela.

   Finalidad: Ejemplo básico de asignación a una variable del resultado de una expresión.. Dificultad Baja.

6. Las ganancias de un determinado producto se reparten entre el diseñador y los tres fabricantes del mismo. Diseñar un programa que pida la ganancia total de la empresa (los ingresos realizados con la venta del producto) y diga cuanto cobran cada uno de ellos, sabiendo que el diseñador cobra el doble que cada uno de los fabricantes.

   El dato de entrada será la ganancia total a repartir. Utilizad el tipo `double` para guardar la ganancia total y

a) variables `double`

b) variables `int`

para guardar las ganancias del diseñador y fabricantes,

Muestre los resultados y analice los resultados obtenidos.

Importante: No debe repetir cálculos ya realizados.

Finalidad: Entender la importancia de no repetir cómputos para evitar errores de programación. Dificultad Baja.

7. Modificar el programa que resuelve el ejercicio 3 sustituyendo los literales que hacen referencia al valor de π por una *constante* llamada `PI`.

   Modifique su valor, recompile y ejecute. ¿Es esta solución mejor que la del ejercicio 3? ¿por qué?.

   Finalidad: Entender la importancia de las constantes. Dificultad Baja.

8. Escriba un programa que lea un distancia en millas (como un real) y muestre la cantidad equivalente en kilómetros. A continuación leerá una distancia en kilómetros (como un real) y mostrará la cantidad equivalente en millas.

   Debe tener en cuenta que 1 milla equivale a `1.609` kilómetros.

   Finalidad: Asignar a una variable el resultado de una expresión. Dificultad Baja.

9. Realizar un programa que nos pida una longitud cualquiera dada en metros. El programa deberá calcular el equivalente de dicha longitud en pulgadas, pies, yardas y millas, y mostrarnos los resultados en pantalla. Para el cálculo, utilice la siguiente tabla de conversión del sistema métrico:

1 pulgada= 25,4 milímetros	1 yarda = 0,9144 metros
1 pie = 30,48 centímetros	1 milla = 1609,344 metros

   Finalidad: Plantear la solución con una doble conversión. Dificultad Baja.

10. Escriba un programa que calcule el consumo de gasolina. Pedirá la distancia recorrida (en kms), los litros de gasolina consumidos y los litros que quedan en el depósito. El programa debe informar el consumo en *km/litro*, los *litros/100 km* y cuantos kilómetros de autonomía le restan con ese nivel de consumo.

    Finalidad: Resolver un problema real sencillo, usando varias sentencias. Dificultad Baja.

11. Queremos construir un programa que simule un juego inspirado en el de los triles (del que procede el nombre de *trilero*). Suponemos que hay dos participantes y cada uno tiene una caja etiquetada con su nombre. Dentro de cada caja hay una cantidad de dinero y el objetivo es intercambiar las cantidades que hay dentro. Por ahora, sólo se pide construir un programa que haga lo siguiente:

- Debe leer desde teclado dos variables `caja_izda` y `caja_dcha`.
- A continuación debe intercambiar sus valores y finalmente, mostrarlos.

Observe que se desea intercambiar el contenido de las variables, de forma que `caja_izda` pasa a contener lo que tenía `caja_dcha` y viceversa. El siguiente código no es válido ya que simplemente engaña al usuario pero las cajas no se quedan modificadas:

```
cout << "La caja izquierda vale " << caja_dcha << "\n";
cout << "La caja derecha vale " << caja_izda;
```

Estaríamos tentados a escribir el siguiente código:

```
caja_izda = caja_dcha;
caja_dcha = caja_izda;
```

pero no funciona correctamente ¿Por qué?

Proponga una solución e impleméntela.

Finalidad: Entender cómo funciona la asignación entre variables. Dificultad Baja.

12. Leed desde teclado tres variables correspondientes a un número de horas, minutos y segundos, respectivamente. Diseñar un algoritmo que calcule las horas, minutos y segundos dentro de su rango correspondiente. Por ejemplo, dadas 10 horas, 119 minutos y 280 segundos, debería dar como resultado 12 horas, 3 minutos y 40 segundos. En el caso de que nos salgan más de 24 horas, daremos también los días correspondientes (pero ya no pasamos a ver los meses, años, etc)

    Como consejo, utilizad los operadores / (representa la división entera cuando los dos argumentos son enteros) y % (representa el resto de la división entera).

    Finalidad: Usar expresiones y variables para no repetir cómputos. Dificultad Media.

13. Calcular el número de segundos que hay entre dos instantes del mismo día. Cada instante se caracteriza por la hora (entre 0 y 23), minuto (entre 0 y 59) y segundo (entre 0 y 59).

    El programa leerá la hora, minuto y segundo del instante inicial, y la hora, minuto y segundo del instante final (supondremos que los valores introducidos son correctos) y mostrará el número de segundos entre ambos instantes.

    Finalidad: Trabajar con expresiones numéricas y algoritmos. Dificultad Media.

14. En atletismo se expresa la rapidez de un atleta en términos de ritmo (*minutos y segundos por kilómetro*) más que en unidades de velocidad (*kilómetros por hora*).

    Escribid dos programas para convertir entre estas dos medidas:

a) El primero leerá el ritmo (minutos y segundos, por separado) y mostrará la velocidad (kilómetros por hora).

b) El segundo leerá la velocidad (kilómetros por hora) y mostrará el ritmo (minutos y segundos).

Finalidad: Trabajar con expresiones numéricas y variables de diferentes tipos. Dificultad Baja.

15. Redactar un algoritmo para calcular la media aritmética muestral y la desviación estándar (o típica) muestral de las alturas de **tres** personas. Éstos valores serán reales (de tipo `double`). La fórmula general para un valor arbitrario de `n` es:

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i \ , \ \sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{X})^2}$$

$\overline{X}$ representa la media aritmética y σ la desviación estándar. Para resolver este problema es necesario usar la función `sqrt` (raíz cuadrada) que se encuentra en la biblioteca `cmath`.

Estas medidas se utilizan mucho en Estadística para tener una idea de la distribución de datos. La media (*mean*, en inglés) nos da una idea del valor central y la desviación típica (*standard deviation*, en inglés) nos da una idea de la dispersión de éstos.

Ejecutad el programa con varios valores y comprobad que el resultado es correcto utilizando una calculadora científica o cualquier calculadora online como por ejemplo la disponible en http://www.disfrutalasmatematicas.com/datos/desviacion-estandar-calculadora.html

Finalidad: Trabajar con expresiones numéricas y con variables para no repetir cómputos. Dificultad Baja.

16. La función gaussiana es muy importante en Estadística. Es una función real de variable real en la que el parámetro μ se conoce como *esperanza* o *media* y σ como *desviación típica* (*mean* y *standard deviation* en inglés, respectivamente). Su definición viene dada por la expresión:

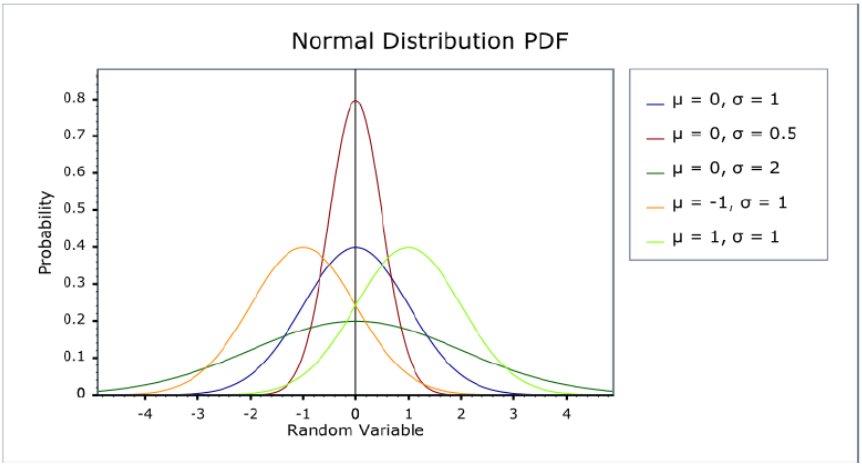
$$\text{gaussiana}(x) = \frac{1}{\sigma\sqrt{2\pi}} \ e^{\left\{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right\}}$$

Realizar un programa que lea los coeficientes reales μ y σ de una función gaussiana. A continuación el programa leerá un valor de abscisa x y se imprimirá el valor que toma la función en x.

Para realizar las operaciones indicadas, debe utilizar las siguientes funciones de la biblioteca `cmath`:

- Para elevar el número e a un valor cualquiera, use la función `exp`. Por ejemplo, para calcular e^8 debería usar la expresión `exp(8)`
- Para calcular la raíz cuadrada, use `sqrt`. Por ejemplo, para calcular $\sqrt{8}$ debería usar la expresión `sqrt(8)`
- Para elevar un número a otro, utilice la función `pow` en la siguiente forma: `pow(base, exponente)`. Por ejemplo, para calcular 2^{10} debería usar la expresión `pow(2,10)`

En la gráfica siguiente pueden verse algunos ejemplos de esta función con distintos parámetros.



Comprobad que los resultados son correctos, usando:
http://danielsoper.com/statcalc3/calc.aspx?id=54
https://www.easycalculation.com/statistics/normal-pdf.php

Finalidad: Trabajar con expresiones numéricas más complejas. Dificultad Media.

17. Escribir un programa que lea un valor entero. Supondremos que el usuario introduce **siempre** un entero de tres dígitos, como por ejemplo 351. Escribid en pantalla los dígitos separados por tres espacios en blanco. Con el valor anterior la salida sería:

```
3   5   1
```

Nota: Más adelante aprenderemos a comprobar si el número introducido tiene tres dígitos, y cómo actuar (pidiendo otro número) en el caso de que no los tenga .

Dificultad Media.

18. Diseñar un programa que lea un carácter (supondremos que el usuario introduce una mayúscula), lo pase a minúscula y lo imprima en pantalla. Hacedlo sin usar las funciones `toupper` ni `tolower` de la biblioteca `cctype`. Para ello, debe considerarse la equivalencia en C++ entre los tipos enteros y caracteres.

Finalidad: Entender la equivalencia entre tipos enteros y de carácter. Dificultad Baja.

19. Supongamos el siguiente código:

```
int entero;
char caracter;

cin >> caracter;
entero = caracter;
```

Supongamos que ejecutamos el código e introducimos el 7 desde teclado. El programa está leyendo una variable de tipo `char`. Por lo tanto, el 7 se interpreta como un carácter y es como si hiciésemos la siguiente asignación:

```
caracter = '7';
entero = caracter;
```

La variable `caracter` almacena internamente el valor 55 (número ASCII del carácter 7). La variable `entero` almacenará, también, el valor 55

Queremos construir un progrrama para asignarle a la variable `entero` el número 7, asociado al dígito representado en la variable `caracter`, es decir, el 7 (no el 55) ¿Cómo lo haría? El programa debe mostrar el resultado.

Nota. La comilla simple para representar un literal de carácter es la que hay en el teclado del ordenador debajo de la interrogación `?`.

Finalidad: Entender la equivalencia entre tipos enteros y de carácter. Dificultad Baja.

20. Dadas las variables `count = 0`, `limit = 10`, `x = 2` e `y = 7`, calcule el valor de las siguientes expresiones lógicas:

```
count == 0 && limit < 20
limit > 20 || count < 5
!(count == 12)
count == 1 && x < y
!( (count < 10 || x < y) && count >= 0 )
(count > 5 && y == 7) || (count <= 0 && limit == 5*x)
!( limit != 10 && z > y )
```

21. Razonar sobre la falsedad o no de las siguientes afirmaciones:

    a) `'c'` es una expresión de caracteres.

    b) 4<3 es una expresión numérica.

    c) (4+3)<5 es una expresión numérica.

    d) `cout << a;` da como salida la escritura en pantalla de una `a`.

e) ¿Qué realiza `cin >> cte`, siendo `cte` una constante entera?

Finalidad: Distinguir entre expresiones de distinto tipo de dato. Dificultad Baja.

22. Indicar si se produce un problema de precisión o de desbordamiento en los siguientes ejemplos indicando cuál sería el resultado final de las operaciones.

a)
```
int chico, chico1, chico2;
chico1 = 123456789;
chico2 = 123456780;
chico = chico1 * chico2;
```

b)
```
long grande;
int chico1, chico2;
chico1 = 123456789;
chico2 = 123456780;
grande = chico1 * chico2;
```

c)
```
double res, real1, real2;
real1 = 123.1;
real2 = 124.2;
res = real1 * real2;
```

d)
```
double resultado, real1, real2;
real1 = 123456789.1;
real2 = 123456789.2;
resultado = real1 * real2;
```

e)
```
double real, otro;
real = 2e34;
otro = real + 1;
otro = otro - real;
```

f)
```
double real, otro;
real = 1e+300;
otro = 1e+200;
otro = otro * real;
```

g)
```
float chico;
double grande;
grande = 2e+150;
chico = grande;
```

Nota. Si se desea ver el contenido de una variable real con `cout`, es necesario que antes de hacerlo, se establezca el número de decimales que se quieren mostrar en pantalla. Hacedlo escribiendo la sentencia `cout.precision(numero_digitos);`, en cualquier sitio del programa antes de la ejecución de `cout << real1 << "," << real2;`. Hay que destacar que al trabajar con reales siempre debemos asumir representaciones aproximadas por lo que no podemos pensar que el anterior valor `numero_digitos` esté indicando un número de decimales con representación exacta.

Finalidad: Entender los problemas de desbordamiento y precisión. Dificultad Media.

23. Escribid una expresión lógica que sea verdadera si una variable de tipo carácter llamada `letra` es una letra minúscula y falso en otro caso.

    Escribid una expresión lógica que sea verdadera si una variable de tipo entero llamada `edad` es menor de 18 o mayor de 65.

Escribid una expresión lógica que nos informe cuando un año es bisiesto. Los años bisiestos son aquellos que o bien son divisibles por 4 pero no por 100, o bien son divisibles por 400.

Nota: Cuando se imprime por pantalla (con `cout`) una expresión lógica que es true, se imprime 1. Si es false, se imprime un 0. En el tema 2 veremos la razón.

Finalidad: Usar expresiones lógicas, muy usadas en el tema 2. Dificultad Baja.

24. Indique qué tipo de dato usaría para representar:

    - Edad de una persona
    - Producto interior bruto de un país. Consultad: http://es.wikipedia.org/wiki/Anexo:Pa%C3%ADses_por_PIB_(nominal)
    - La cualidad de que un número entero sea primo o no.
    - Estado civil (casado, soltero, separado, viudo)
    - Sexo de una persona (hombre o mujer exclusivamente)

    Finalidad: Saber elegir adecuadamente un tipo de dato, atendiendo a la información que se quiere representar. Dificultad Media.

25. El precio final de un automóvil para un comprador es la suma total del costo del vehículo, del porcentaje de ganancia de dicho vendedor y del I.V.A. Diseñar un algoritmo para obtener el precio final de un automóvil sabiendo que el porcentaje de ganancia de este vendedor es del 20 % y el I.V.A. aplicable es del 16 %.

    Dificultad Baja.

26. Cread un programa que lea un valor de temperatura expresada en grados Celsius y la transforme en grados Fahrenheit. Para ello, debe considerar la fórmula siguiente:

    ```
    Grados Fahrenheit = (Grados Celsius * 180 / 100) + 32
    ```

    Buscad en Internet el por qué de dicha fórmula.

    Dificultad Baja.

27. Cread un programa que lea las coordenadas de dos puntos $P_1 = (x_1, y_1)$ y $P_2 = (x_2, y_2)$ y calcule la distancia euclídea entre ellos:

$$d(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

    Dificultad Baja.

28. Declarar las variables necesarias y traducir las siguientes fórmulas a expresiones válidas del lenguaje C++.

    a) $\dfrac{1+\frac{x^2}{y}}{\frac{x^3}{1+y}}$

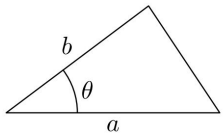
    b) $\dfrac{1+\frac{1}{3}\sin h-\frac{1}{7}\cos h}{2\,h}$

    c) $\sqrt{1+\left(\dfrac{e^x}{x^2}\right)^2}$

Algunas funciones de `cmath`
$\mathrm{sen}(x) \longrightarrow$ `sin(x)`
$\cos(x) \longrightarrow$ `cos(x)`
$x^y \longrightarrow$ `pow(x, y)`
$\ln(x) \longrightarrow$ `log(x)`
$\mathrm{e}^x \longrightarrow$ `exp(x)`

    Dificultad Baja.

29. El área A de un triángulo se puede calcular a partir del valor de dos de sus lados, a y b, y del ángulo θ que éstos forman entre sí con la fórmula $A = \frac{1}{2}ab\,sin(\theta)$. Construid un programa que pida al usuario el valor de los dos lados (en centímetros), el ángulo que éstos forman (en grados), y muestre el valor del área.

    Tened en cuenta que el argumento de la función `sin` va en radianes por lo que habrá que transformar los grados del ángulo en radianes.

    Dificultad Baja.

30. Los compiladores utilizan siempre el mismo número de bits para representar un tipo de dato entero (este número puede variar de un compilador a otro). Por ejemplo, 32 bits para un `int`. Pero, realmente, no se necesitan 32 bits para representar el 6, por ejemplo, ya que bastarían 3 bits:

    $$6 = 1 * 2^2 + 1 * 2^1 + 0 * 2^0 \equiv 110$$

    Se pide crear un programa que lea un entero n, y calcule el mínimo número de dígitos que se necesitan para su representación. Para simplificar los cómputos, suponed que sólo queremos representar valores enteros positivos (incluido el cero). Consejo: se necesitará usar el logaritmo en base 2 y obtener la parte entera de un real (se obtiene tras el truncamiento que se produce al asignar un real a un entero)

    Dificultad Media.