using namespace

si no usas using name space se indica donde buscar lo con std : :

std: :cout << "½Hola a todos!" << std: :endl;

Pag 6

El valor de la constante puede ser desconocido hasta que se ejecute el código

Pag 7

En vez de el código objeto llamar a la función, se introduce en el código objeto

pag 11

Una clase es una abstracción de datos y operadores

Pag 12

const no cambia el contenido, son funciones consultoras

Pag 16

+= es un operador de sobrecarga por ejemplo cuando lo usamos para concatenar

toupper función para poner en mayúsculas

Pag 18

Consultor – consulta pero no modifica

contructor – cnstruye el objeto

tiempo devuelve el tiempo transcurrido desde que está en marcha

Pag 19

La “L” se usa cuando es un número grande

EJERCICIOS

1. Considere la siguiente función:

Texto, Carta

Descripción generada automáticamente

* 1. ¿Cuál es el resultado de las siguientes llamadas?

muestra\_temperatura(10); --> Muestra por pantalla baja

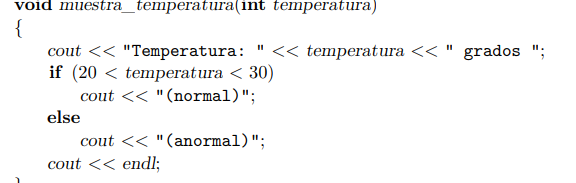
muestra\_temperatura(27); --> Muestra por pantalla normal

muestra\_temperatura(40); --> No muestra nada por pantalla

* 1. ¿Cree que coincide con las expectativas del programador? ¿A qué se debe?

No, viendo el programa, tendría mas lógica que si temperatura > 20 y temperatura < 30 mostrar temperatura normal, y sí es mayor que 30 mostrara temperatura alta

1. Considere la siguiente función:



Explique el resultado de las siguientes llamadas a la función:

muestra\_temperatura(10); --> Muestra por pantalla Temperatura: 10 grados anormal

muestra\_temperatura(27); --> Muestra por pantalla Temperatura: 27 grados normal

muestra\_temperatura(40); --> Muestra por pantalla Temperatura: 40 grados anormal

1. ¿Qué problema hay con el siguiente programa?

Texto, Carta

Descripción generada automáticamente

Comente, al menos, dos posibles soluciones.

El problema está en

1. El siguiente programa se compone de una cabecera y dos módulos:

Texto

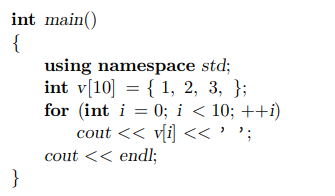
Descripción generada automáticamente

¿Ve algún problema? Si es así, explique cómo solucionarlo con el mínimo esfuerzo. Realice una crítica razonada de este estilo de programación.

La variable visitas esta declarada en la cabecera de forma global por lo que al incluirla en los módulos auxiliares se importar en ambos y generará un error ya que no se puede definir la misma variable dos veces.

Declarar variables globales no es una buena práctica a la hora de programar ya que puede llegar a errores como este o errores lógicos en el código que son difíciles de detectar.

1. ¿Hay algún problema con el siguiente programa? Si lo hay, explique cuál. Si no, explique qué salida produce.



La inicialización del vector es incorrecta, detrás de 3, deberíamos introducir otro valor del vector o en caso de que no queramos quitar la coma, pero no podemos dejarlo en blanco.

Por otro lado en el bucle mostramos los valores del vector del 0 al 9, pero no hemos inicializado las 10 posiciones del vector, esto provoca que el programa mustre lo que hay guardado en esas posiciones de memoria (basura), por ejemplo 1, 2, 3, dwf1. 34dc, 2d2g, … Para solucionar esto debemos inicializar los 10 elemento o mostrar solo los que inicializamos.

1. Escriba una función que se expanda en línea y, utilizando referencias, intercambie los valores de sus tres parámetros enteros rotándolos a la izquierda. Sobrecargue esta función utilizando punteros. Después compare y contraste las dos funciones y la manera de llamarlas.

Void funcion(int a, int b, int c) {

Int aux = b;

b = c;

c = a;

a = aux;

}

Void función\_punteros(int \*a, int \*b, int \*c) {

Int \*aux = \*b;

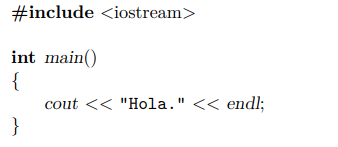
\*b = \*c;

\*c = a;

\*a = \*aux;

}

1. ¿Qué problema hay con el siguiente programa?



Explique, al menos, tres formas distintas de resolverlo.

cout y endl so funciones de la librería estándar y se lo tenemos que indicar al compilador de alguna de estas tres maneras:

* Std::cout << “Hola.” << std::endl;

Con std le indicamos que pertenece a la librería estandar

* Using namespace sdt;

De esta forma no hará falta poner std::cout cada vez que mostremos algo por pantalla, valdrá simplemente con cout << “”

* Endl también lo podremos sustituir por \n al final de la cadena std::cout << “Hola. \n”-

1. Escriba un programa que calcule el cuadrado de un double leído de la entrada estándar e imprima el número de veces que puede repetir dicho cálculo durante un segundo. Emplee una función que se expanda en línea para obtener el cuadrado de un número.

#include <iostream>

#include <chrono>

// Función para calcular el cuadrado de un número (se expande en línea)

inline double calcularCuadrado(double num) {

return num \* num;

}

int main() {

double numero;

int iteraciones = 0;

std::cout << "Ingresa un número: ";

std::cin >> numero;

auto inicio = std::chrono::high\_resolution\_clock::now();

// Cálculo del cuadrado y conteo de iteraciones en un segundo

while (true) {

double cuadrado = calcularCuadrado(numero);

iteraciones++;

auto actual = std::chrono::high\_resolution\_clock::now();

auto duracion = std::chrono::duration\_cast<std::chrono::microseconds>(actual - inicio).count();

if (duracion >= 1000000) {

// Si ha pasado un segundo, salimos del bucle

break;

}

}

std::cout << "El cuadrado de " << numero << " es " << calcularCuadrado(numero) << std::endl;

std::cout << "Número de iteraciones en un segundo: " << iteraciones << std::endl;

return 0;

}