Algunos de ustedes están teniendo problemas compilando el código que escribí en el pizarrón. A continuación presento un código de ejemplo de lo que vimos en clase y señalo la línea en donde no compila.

```
#include <iostream>

struct nodo {
   int valor;
   nodo* izq = nullptr;
   nodo* der = nullptr;
};

void inserta(nodo*& p, int v) {
   if (p == nullptr) {
      p = new nodo{v};             // error en esta línea
   } else if (v < p->valor) {
      inserta(p->izq, v);
   } else if (v > p->valor) {
      inserta(p->der, v);
   }
}

void orden(nodo* p) {
   if (p == nullptr) {
      return;
   }

   orden(p->izq);
   std::cout << p->valor << "\n";
   orden(p->der);
}

int main( ) {
   nodo* raiz = nullptr;
   inserta(raiz, 5);
   inserta(raiz, 7);
   inserta(raiz, 3);
   inserta(raiz, 8);
   inserta(raiz, 1);

   orden(raiz);
}
```

El problema es la versión de C++. Tendré que dar la razón técnica para que la explicación quede completa, pero tal vez sea un poco confusa.

Un `struct` se considera un *agregado* si no tiene constructores ni ninguna característica especial de programación orientada a objetos. En particular, el siguiente `struct` se considera un agregado en C++11:

```
struct nodo {
   int valor;
   nodo* izq;
   nodo* der;
};
```

Sin embargo, al especificar valores por defecto deja de ser un agregado en C++11 (lo sigue siendo en C++17):

```
struct nodo {
   int valor;
   nodo* izq = nullptr;      // ya no es un agregado en C++11, sí lo es en C++17
   nodo* der = nullptr;      // ya no es un agregado en C++11, sí lo es en C++17
};
```

Lo malo es que en este caso, la notación `new nodo{v}` sólo funcionará correctamente si `nodo` es un agregado. Una manera de que compile es que se pasen a C++17 (tendrían que actualizar su compilador) pero desafortunadamente OmegaUp sólo tiene soporte para C++11. Entonces hay dos alternativas:

1) Quitar los valores por defecto del `struct` para que sí se considere un agregado en C++11. Esto no causa problemas en el ejemplo visto en clase, porque al usar la notación de llaves y sólo especificar `v`, los demás miembros se llenan automáticamente con el *cero* del tipo (para apuntadores, con `nullptr`).

2) Inicializar el nodo sin usar la notación `new nodo{v}`. Tendrían que cambiar
   ```
   p = new nodo{v};
   ```
   por
   ```
   p = new nodo;
   p->valor = v;
   ```

Creo que la primera opción es mejor. En pocas palabras, es mi culpa que lo que escribí en el pizarrón no les compile en C++11, pero desafortunadamente C++ tiene reglas muy raras :(

https://en.cppreference.com/w/cpp/language/aggregate_initialization