Pontificia Universidad
JAVERIANA
Cali

Uso del debugger

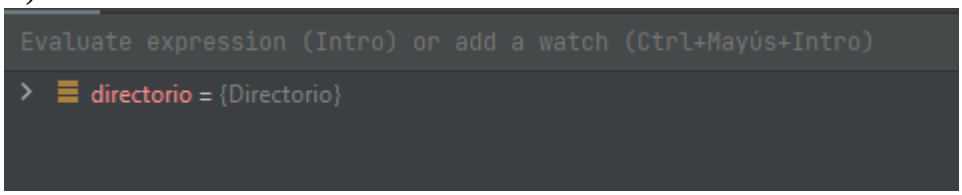
Jean Carlos Santacruz Arredondo

Ingeniería de Sistemas

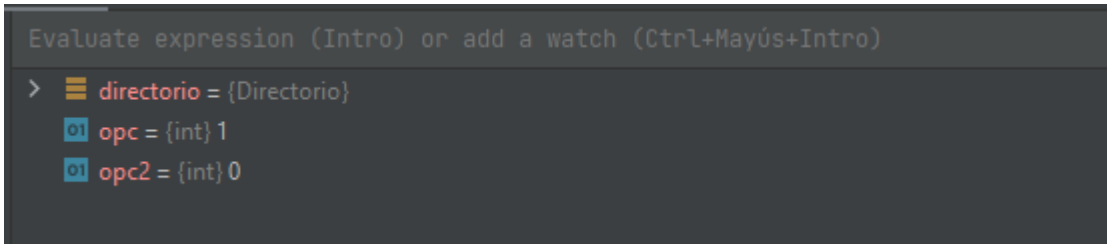
programación orientada a objetos

Tercer semestre

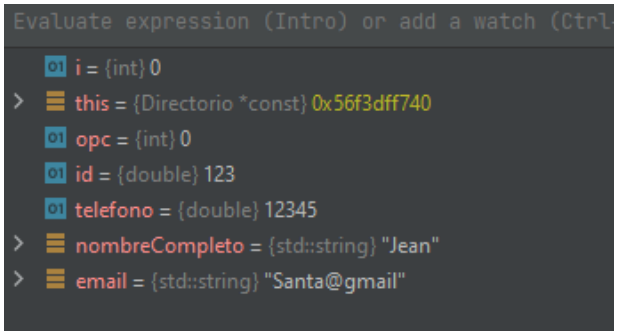
1- A) El método utilizado fue el de crear cliente voy a manipular el telefono
B) Main:



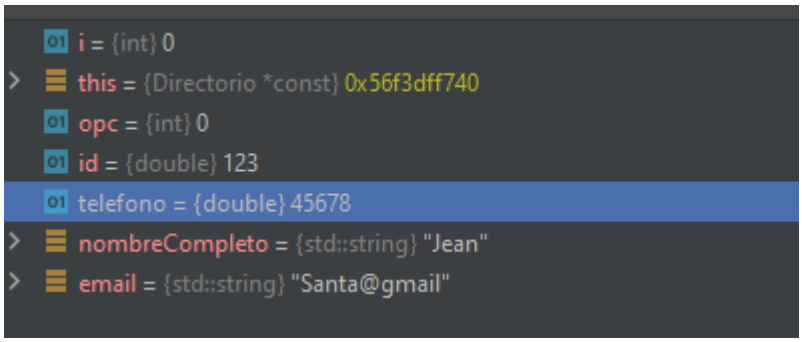
Menu:



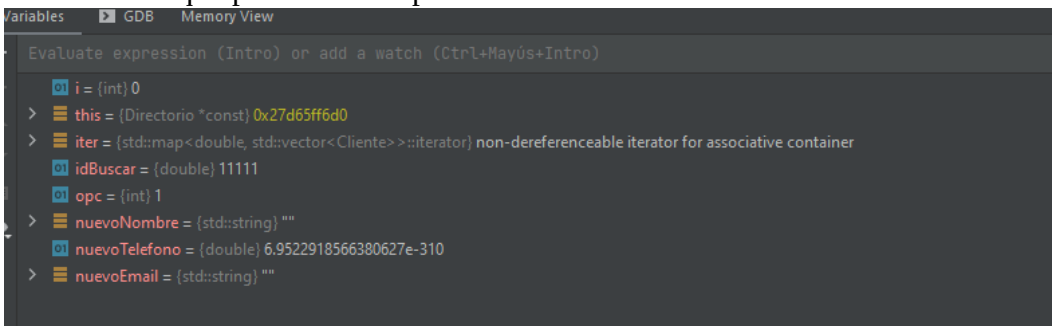
Propietario:



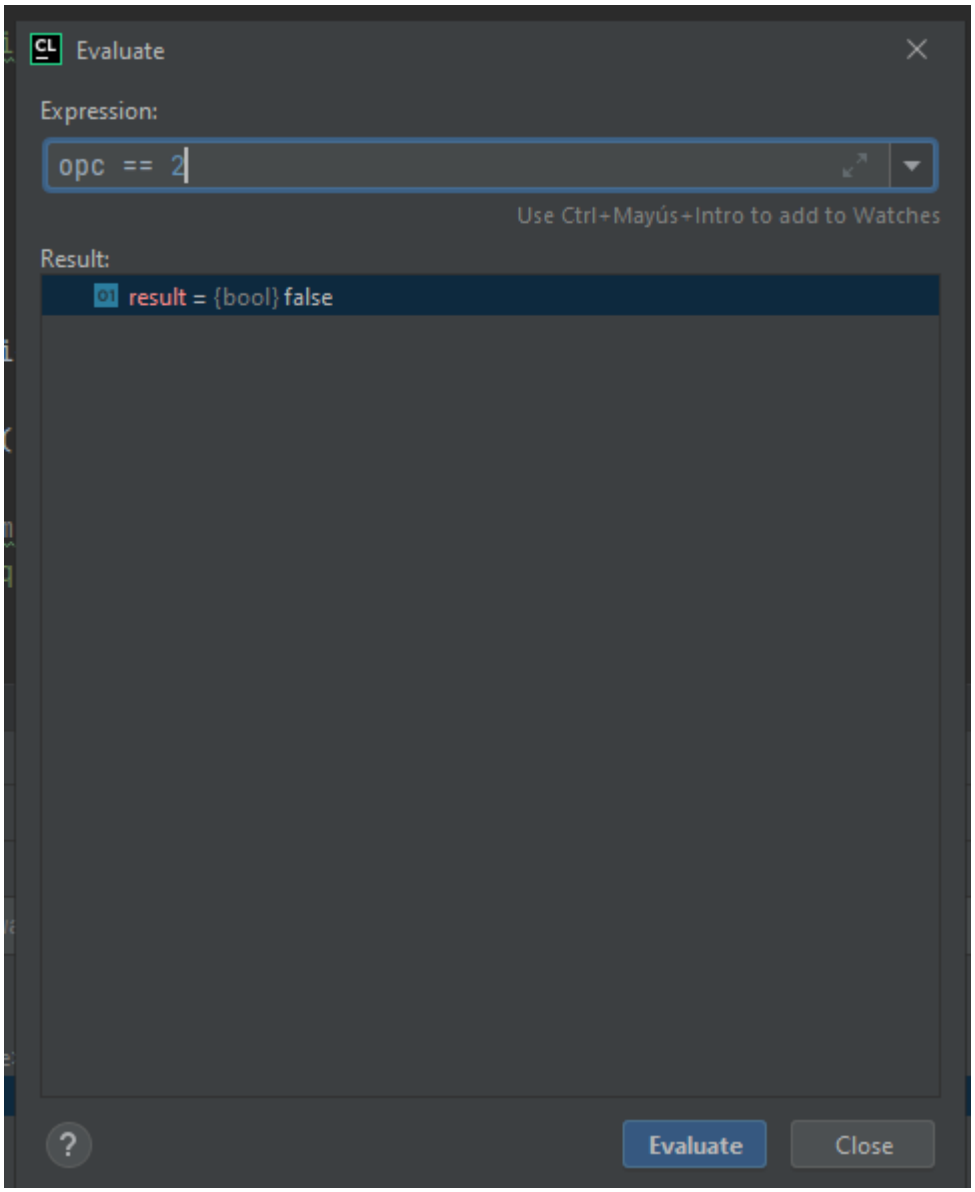
2-



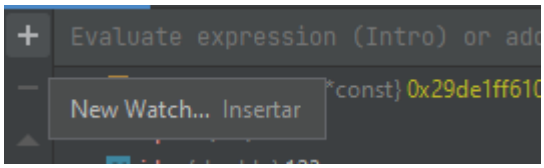
3- BreakPoint Condicional, lo incorpore en la función modificar clientes, con la condición de que pare cuando opc == 1

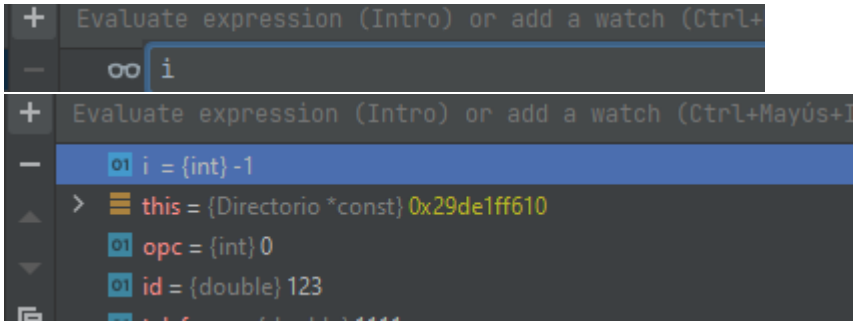


4-


Evaluate
Expression:
opc == 2
Use Ctrl+Mayús+Intro to add to Watches
Result:
result = {bool} false
Evaluate
Close

5- S


Evaluate expression (Intro) or ad
New Watch... Insertar



6-

step over (F8) y la funcionalidad step into (F7), el step into nos permite ver mas a detalle cada cosa del código, es como un paso a paso, por ejemplo en la funcionalidad imprimir clientes tengo un ciclo que itera el tamaño del vector clientes y cada vez que itera se topa con 4 funciones las cuales son gets que me traen la información del cliente, con el step into va y me muestra que hace cada función y todo el recorrido que hace para el valor que retorna, mientras con el step over solo me muestra el valor que retorna pero no el proceso realizado, en conclusión step into me muestra el paso a paso y step over solo el resultado.

```
void Directorio::imprimirInformacionClientes()
{
    cout << "Informacion de nuestros clientes: \n";
    for (int i = 0; i < clientes.size(); i++)
    {
        cout << "Identificacion: " << clientes[i].getIdentificacion() << "\n";
        cout << "Nombre: " << clientes[i].getNombre() << "\n";
        cout << "Email: " << clientes[i].getEmail() << "\n";
        cout << "Telefono: " << clientes[i].getTelefono() << "\n";
    }
}
```