

 <p>ESCUELA POLITÉCNICA NACIONAL Programación I</p>	<p>LABORATORIO No. 2</p>
<p>NOMBRE: Danny Paúl Ponce Guacalés.</p>	<p>FECHA DE ENTREGA: 17/06/2025</p>

PRÁCTICA DE LABORATORIO No. 2

TEMA: Clean Code

1. OBJETIVOS

- Aplicar los principios del clean code a un programa en C++, mediante la reestructuración de un código, con el fin de poder mejorar la legibilidad y hacer al código más fuerte ante cambios, así como preverlo de errores
- Implementar los conocimientos adquiridos en clase, durante el primer bimestre a fin de arreglar el código proporcionado, dándole un menú interactivo y solicitando que se ingresen los datos, con el fin de garantizar resultados más precisos y consistentes

2. TEORÍA

La práctica 2 de laboratorio se enfoca en aplicar los principios del clean code a un programa en C++ destinado a la conversión de unidades. Los principios usados para tener un código limpio son la organización del código, aplicación de nombres descriptivos para las variables.

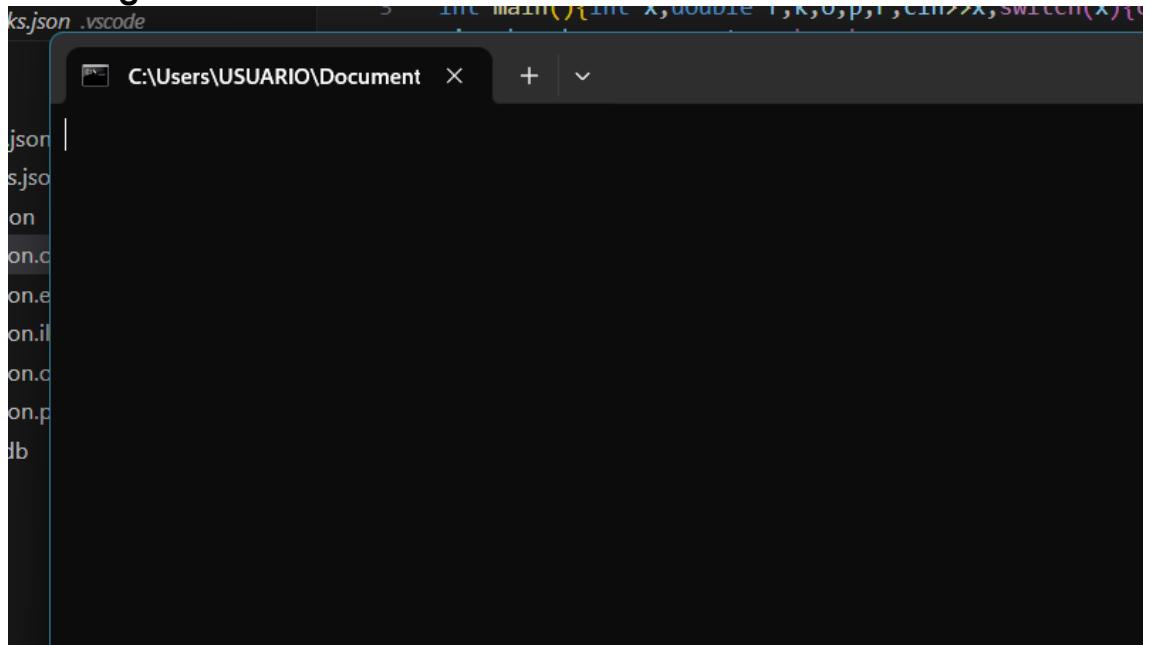
El programa inicial que fue proporcionado funciona, sin embargo, carece de una estructura al punto que ni siquiera solicitar datos ni tener un menú en el cual seleccionar qué hacer. Al finalizar esta práctica se espera un código que mantenga el funcionamiento, pero que este sea más ordenado, y que tenga un menú interactivo, así como amigable con los usuarios que lo usen.

3. DESARROLLO

1. Se copió y pego el código proporcionado en una IDE en este caso Visual Studio Code usando el compilador MSVC; podemos observar que el código está totalmente desordenado e incluso ilegible.
2. Compilamos el código y observamos que la salida de este nos da la terminal vacía, esperando que ingrese el usuario un dato sin embargo no se sabe qué se debe ingresar, tal como se muestra

 <p>ESCUELA POLITÉCNICA NACIONAL Programación I</p>	<p>LABORATORIO No. 2</p>
<p>NOMBRE: Danny Paúl Ponce Guacalés.</p>	<p>FECHA DE ENTREGA: 17/06/2025</p>

en la **Figura 1**.



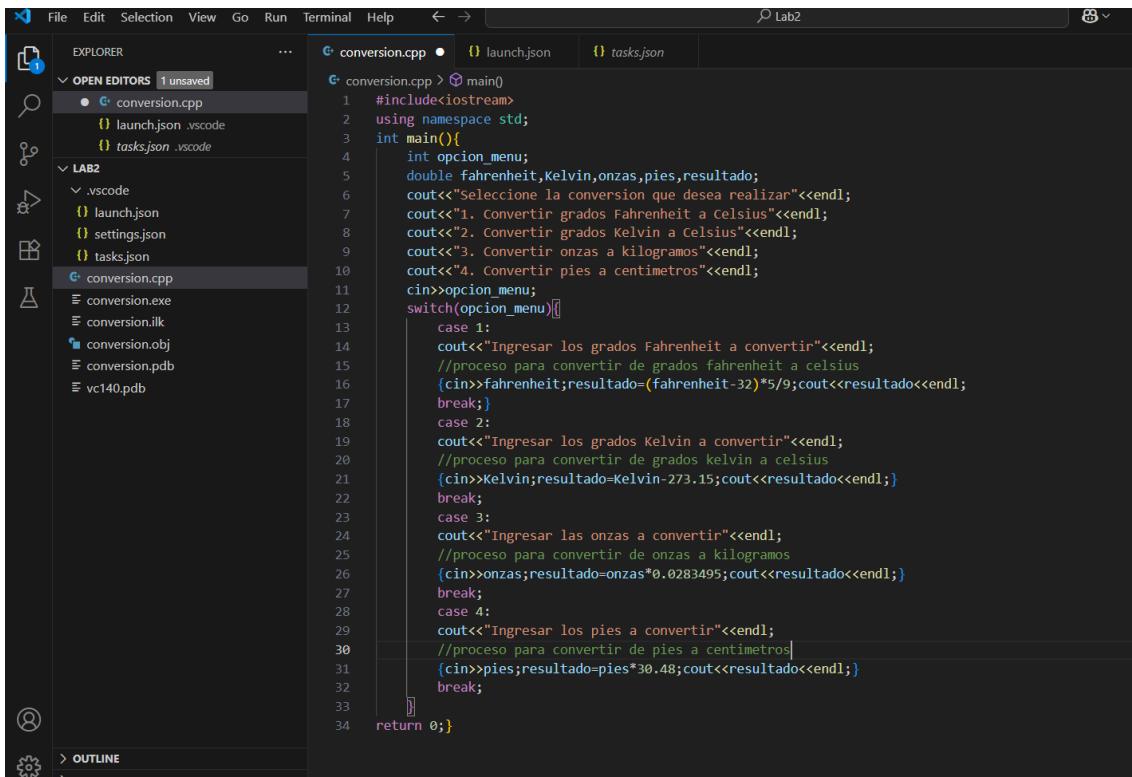
```
ks.json .vscode
C:\Users\USUARIO\Document x + v
json
s.jso
on
on.c
on.e
on.il
on.c
on.p
db
```

Figura 1. Salida del código proporcionado

3. Al ver esto procedemos a trabajar en dar un menú interactivo que le diga al usuario que es aquello que debe seleccionar, así como pedir que ingrese un valor para la transformación.
4. Cambiamos los nombres de las variables a unos más específicos y que se pueda saber que dato es el que se está usando.
5. Finalmente organizamos de mejor manera el código y agregamos los comentarios necesarios para que se pueda leer fácilmente, tal como se muestra en la **Figura 2**. De esta manera cumplimos con los principios del clean code, investigados en freeCodeCamp (2020) y los aprendidos en clase.

4. PROGRAMA IMPLEMENTADO

 <p>ESCUELA POLITÉCNICA NACIONAL Programación I</p>	LABORATORIO No. 2
NOMBRE: Danny Paúl Ponce Guacalés.	FECHA DE ENTREGA: 17/06/2025



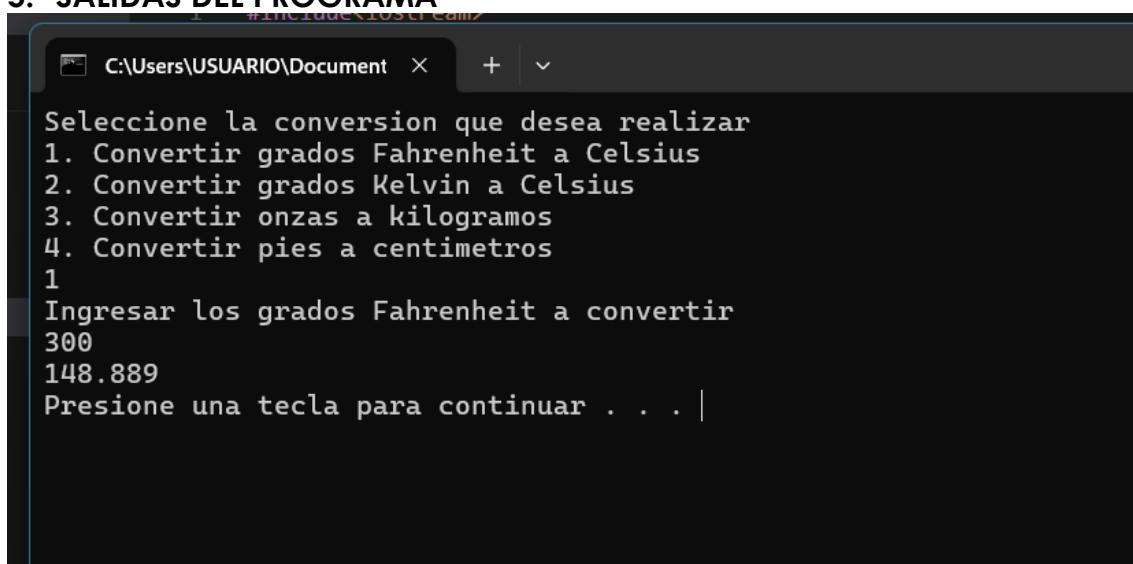
```

File Edit Selection View Go Run Terminal Help < > Lab2
EXPLORER OPEN EDITORS 1 unsaved
conversion.cpp .vscode launch.json tasks.json
conversion.cpp main()
1 #include<iostream>
2 using namespace std;
3 int main(){
4     int opcion_menu;
5     double fahrenheit,Kelvin,onzas,pies,resultado;
6     cout<<"Seleccione la conversion que desea realizar"<<endl;
7     cout<<"1. Convertir grados Fahrenheit a Celsius"<<endl;
8     cout<<"2. Convertir grados Kelvin a Celsius"<<endl;
9     cout<<"3. Convertir onzas a kilogramos"<<endl;
10    cout<<"4. Convertir pies a centimetros"<<endl;
11    cin>>opcion_menu;
12    switch(opcion_menu){
13        case 1:
14            cout<<"Ingresar los grados Fahrenheit a convertir"<<endl;
15            //proceso para convertir de grados fahrenheit a celsius
16            {cin>>fahrenheit;resultado=(fahrenheit-32)*5/9;cout<<resultado<<endl;}
17            break;
18        case 2:
19            cout<<"Ingresar los grados Kelvin a convertir"<<endl;
20            //proceso para convertir de grados kelvin a celsius
21            {cin>>Kelvin;resultado=Kelvin-273.15;cout<<resultado<<endl;}
22            break;
23        case 3:
24            cout<<"Ingresar las onzas a convertir"<<endl;
25            //proceso para convertir de onzas a kilogramos
26            {cin>>onzas;resultado=onzas*0.0283495;cout<<resultado<<endl;}
27            break;
28        case 4:
29            cout<<"Ingresar los pies a convertir"<<endl;
30            //proceso para convertir de pies a centimetros
31            {cin>>pies;resultado=pies*30.48;cout<<resultado<<endl;}
32            break;
33    }
34    return 0;
}

```

Figura 2. En la imagen se muestra el código final, una vez aplicados los principios de clean code y añadido un menú interactivo.

5. SALIDAS DEL PROGRAMA



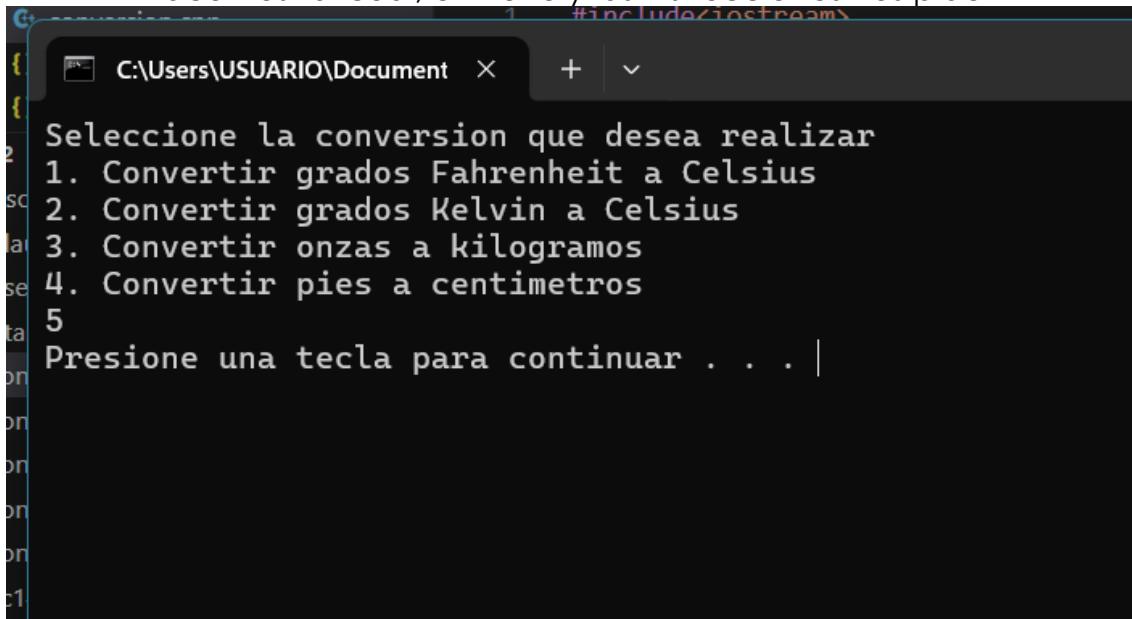
```

C:\Users\USUARIO\Document X + ▾
Seleccione la conversion que desea realizar
1. Convertir grados Fahrenheit a Celsius
2. Convertir grados Kelvin a Celsius
3. Convertir onzas a kilogramos
4. Convertir pies a centimetros
1
Ingresar los grados Fahrenheit a convertir
300
148.889
Presione una tecla para continuar . . .

```

 ESCUELA POLITÉCNICA NACIONAL Programación I	LABORATORIO No. 2
NOMBRE: Danny Paúl Ponce Guacalés.	FECHA DE ENTREGA: 17/06/2025

Figura 3. La imagen nos muestra la salida del programa cuando hacemos tal cual, el menú y las instrucciones nos piden



```

C:\Users\USUARIO\Document x + v
1. #include<iostream>
2. Seleccione la conversion que desea realizar
3.   1. Convertir grados Fahrenheit a Celsius
4.   2. Convertir grados Kelvin a Celsius
5.   3. Convertir onzas a kilogramos
6.   4. Convertir pies a centimetros
7.   5
8. Presione una tecla para continuar . . .

```

Figura 4. La imagen nos muestra la salida del programa cuando ponemos un dato distinto al solicitado, en este caso el programa se cierra gracias al break dentro del código.

6. CONCLUSIONES

- El reescribir el código aplicando principios de clean code no solo mejora la legibilidad y mantenibilidad del programa, sino que también facilita la detección y corrección de errores, ya que es mucho más intuitivo y legible.
- Un menú interactivo bien diseñado, con mensajes claros y una estructura lógica, mejora significativamente la experiencia del usuario, demostrando la importancia de combinar funcionalidad con usabilidad en los programas informáticos

7. REFERENCIAS BIBLIOGRÁFICAS

 <p>ESCUELA POLITÉCNICA NACIONAL Programación I</p>	<p>LABORATORIO No. 2</p>
<p>NOMBRE: Danny Paúl Ponce Guacalés.</p>	<p>FECHA DE ENTREGA: 17/06/2025</p>

freeCodeCamp. (2020, 2 enero). *How to Write Clean Code in C++*.

freeCodeCamp.org. <https://www.freecodecamp.org/news/how-to-write-clean-code-in-c/>