|  |
| --- |
|  |
| CALCULATOR – GROUP4 |

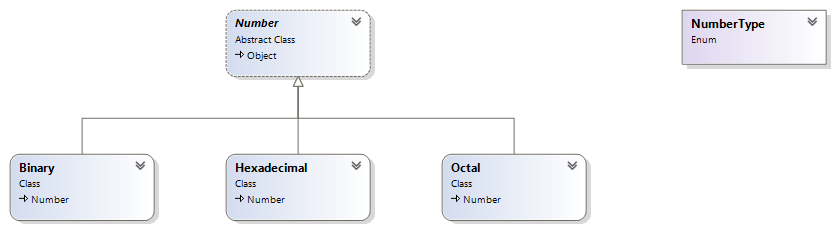
**DIVYA MURUGANANTHAM, STUDENT ID:1857326**

**FAITH EDAFETANURE-IBEH, STUDENT ID:1852320**

**ANVITHA EDA, STUDENT ID:1820715**

Below are the lists of classes created for CALCULATOR project.

**NUMBERS PACKAGE:**

****

**Number – Abstract class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has abstract methods to assign the input value from the user and get the processed value after operations/conversions.

**Binary.cs – Derived class**

It has the fields Bits and Signed properties with constructors to initialize those properties. It also has defined methods to assign the input value from the user and get the processed value after operations/conversions for binary value.

**Hexadecimal.cs – Derived class**

It has the fields Bits and Signed properties with constructors to initialize those properties. It also has defined methods to assign the input value from the user and get the processed value after operations/conversions for hexadecimal value.

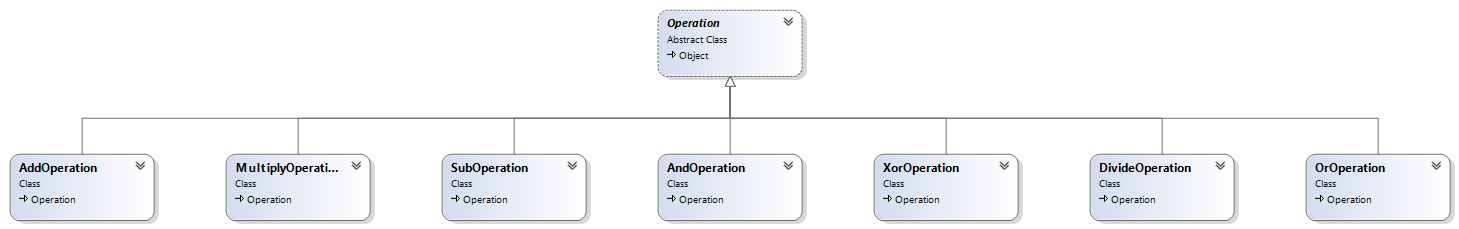
**Octal.cs – Derived class**

It has the fields Bits and Signed properties with constructors to initialize those properties. It also has defined methods to assign the input value from the user and get the processed value after operations/conversions for octal value.

**Enum number types**

It stores all the number types: Octal, Hexadecimal and Binary

**OPERATIONS PACKAGE:**

****

**Operation – Abstract class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has abstract methods to calculate those input value entered by the user.

**AddOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition to calculate the addition of values entered by the user.

**SubOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition to calculate the subtraction of values entered by the user.

**MultiplyOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition to calculate the multiplication of values entered by the user.

**DivideOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition to calculate the division of values entered by the user.

**AndOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition of AND operation for the values entered by the user.

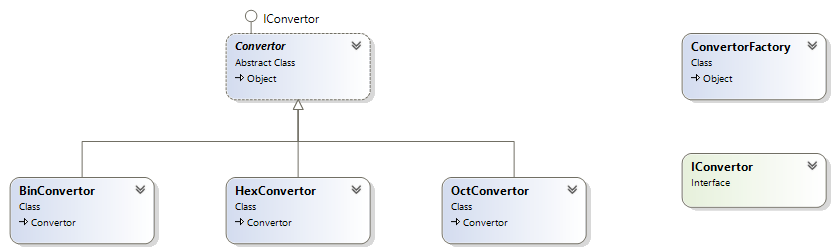
**OrOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition of OR operation for the values entered by the user.

**XorOperation.cs – Derived class**

It has the fields \_bits and \_signed with constructors to initialize those fields. It also has method definition of XOR operation for the values entered by the user.

**CONVERTORS PACKAGE:**

****

**Convertor.cs – Abstract class**

It has abstract methods to convert the given number types to octal/hexadecimal/binary.

**ConvertorFactory.cs**

This class has the convert method to convert the number to the respective numbertype selected by the user.

**IConvertor.cs - Interface**

This interface class has the list of methods present in the convertor.cs

**HexConvertor.cs – Derived class**

This class has a readonly private field \_hex and dictionary to convert from hexadecimal to octal or binary or octal.

**OctConvertor.cs – Derived class**

This class has a readonly private field \_hex and dictionary to convert from octal to hexadecimal or binary or octal.

**BinConvertor.cs – Derived class**

This class has a readonly private field \_hex and dictionary to convert from binary to octal or hexadecimal or octal.

**CALCULATORFORM PACKAGE:**

**Program.cs**

This class acts as a calculator form which prompts input from the user for the operations and conversions and outputs the result.

**TASK COMPLETED IN CALCULATOR PROJECT:**

**Operations:**

1. Add operation is working for binary, hexadecimal and octal.
2. Sub operation is working for binary, hexadecimal and octal.
3. Multiplication operation is working for binary, hexadecimal and octal.
4. Division operation is working for binary, hexadecimal and octal.
5. And operation is working for binary, hexadecimal and octal.
6. Or operation is working for binary, hexadecimal and octal.
7. Xor operation is working for binary, hexadecimal and octal.

**Conversions:**

1. Conversion from binary to hexadecimal is working.
2. Conversion from binary to octal is working.
3. Conversion from binary to binary is working.
4. Conversion from hexadecimal to binary is working.
5. Conversion from hexadecimal to octal is working.
6. Conversion from hexadecimal to hexadecimal is working.
7. Conversion from octal to binary is working.
8. Conversion from octal to hexadecimal is working.
9. Conversion from octal to octal is working.

**RESTRICTIONS IN CALCULATOR PROJECT:**

Below are the restrictions in which calculator project is not working.

1. Calculator Form takes only one input at a time and exits after completion. Sentinel controlled loop is not working.
2. Exception is not completed for Add and Sub operations.