A.1: Difference between Abstraction and Encapsulation in C++

Abstraction

Abstraction is the concept of object-oriented programming that "shows" only essential attributes and "hides" unnecessary information. The main purpose of abstraction is hiding the unnecessary details from the users. Abstraction is selecting data from a larger pool to show only relevant details of the object to the user. It helps in reducing programming complexity and efforts. It is one of the most important concepts of OOPs.

Example of Abstraction:

```cpp
#include <iostream>
using namespace std;

class Summation {
private:
    // private variables
    int myNum1, myNum2, myNum3
public:
    void sum(int inNum1, int inNum2)
    {
        myNum1 = inNum1;
        myNum2 = inNum2;
        myNum3 = myNum1 + myNum2;
        cout << "Sum of the two number is : " << myNum3 << endl;
    }
};
int main()
```
In this case the variables myNum1, myNum2 and myNum3 are private, thereby inaccessible to any code other than the
class Summation. In this example the variables are set to values passed in as arguments to the sum method. This is not
a very true example - often the values would NOT be set just before being used like this, but it shows the reality of the
implementation.

Output:

| Sum of the two number is: 9 |

---

**Encapsulation**

Encapsulation is a method of making a complex system easier to handle for end users. The user need not worry about
internal details and complexities of the system. Encapsulation is a process of wrapping the data and the code, that
operate on the data into a single entity. You can assume it as a protective wrapper that stops random access of code
defined outside that wrapper.

```cpp
#include <iostream>
using namespace std;

class EncapsulationExample {

private:
    // we declare a as private to hide it from outside
    int number1;

public:
    // set() function to set the value of a
    void set(int input1) {
        number1 = input1;
    }

    // get() function to return the value of a
    int get() {
        return number1;
    }

};

// main function
```
int main()
{
    EncapsulationExample myInstance;
    myInstance.set(10);
    cout << myInstance.get() << endl;
    return 0;
}

In the this program, the variable number1 is made private so that this variable can be accessed and manipulated only by using the methods get() and set() that are present within the class. Therefore we can say that, the variable a and the methods set() as well as get() have bound together that is encapsulation. There is nothing special about the method names "get()" or "set()" - there may be other methods that manipulate the variable number1...all together this is called encapsulation.

Output:

10

Difference between Abstraction and Encapsulation:

<table>
<thead>
<tr>
<th>ABSTRACTION</th>
<th>ENCAPSULATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Abstraction is the process or method of gaining the information.</td>
<td>While encapsulation is the process or method to contain the information.</td>
</tr>
<tr>
<td>2. In abstraction, problems are solved at the design or interface level.</td>
<td>While in encapsulation, problems are solved at the implementation level.</td>
</tr>
<tr>
<td>3. Abstraction is the method of hiding the unwanted information.</td>
<td>Whereas encapsulation is a method to hide the data in a single entity or unit along with a method to protect information from outside.</td>
</tr>
<tr>
<td>4. We can implement abstraction using abstract class and interfaces.</td>
<td>Whereas encapsulation can be implemented using by access modifier i.e. private, protected and public.</td>
</tr>
<tr>
<td>5. In abstraction, implementation complexities are hidden using abstract classes and interfaces.</td>
<td>While in encapsulation, the data is hidden using methods of getters and setters.</td>
</tr>
<tr>
<td>6. The objects that help to perform abstraction are encapsulated.</td>
<td>Whereas the objects that result in encapsulation need not be abstracted.</td>
</tr>
</tbody>
</table>

Adapted from:
"Difference between Abstraction and Encapsulation in C++" by MKS075, Geeks for Geeks is licensed under CC BY-SA 4.0