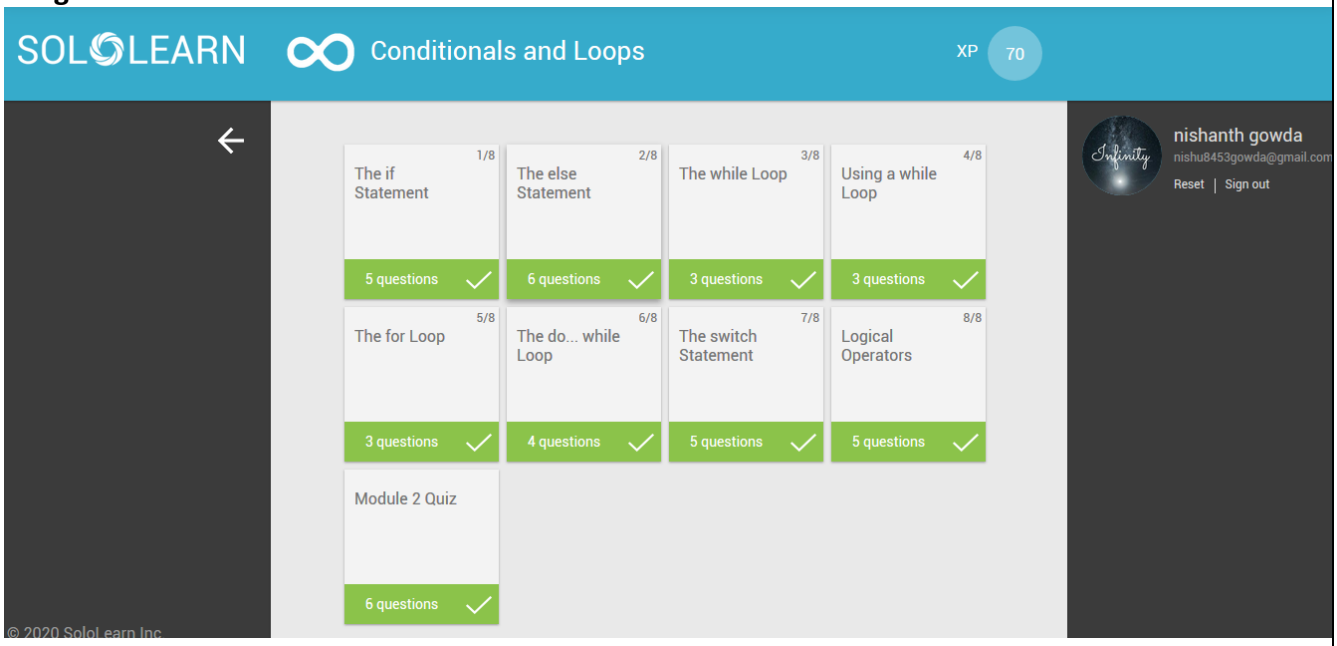
DAILY ASSESSMENT FORMAT

Date:	**22/June/2020**	**Name:**	**nishanth**
Course:	**C++ programming**	**USN:**	**4al17ec063**
Topic:	**1.condition and loop**	**Semester & Section:**	**6th b**
GitHub Repository:	**nishanthvr**		

FORENOON SESSION DETAILS

Image of session



Decision Making

The **if** statement is used to execute some code if a condition is true.

Syntax: if (condition) {
statements
}
The **condition** specifies which expression is to be evaluated. If the condition is true, the statements in the curly brackets are executed.
If the condition is **false**, the statements are simply ignored, and the program continues to run after the if statements body.

Relational Operators

Additional relational operators:

Operator	Description	Example	
>=	Greater than or equal to	7 >= 4	True
<=	Less than or equal to	7 <= 4	False
==	Equal to	7 == 4	False
!=	Not equal to	7 != 4	True

Example:

```
if (10 == 10) {
cout << "Yes";
}

// Outputs "Yes"
```

The else Statement

An **if** statement can be followed by an optional **else** statement, which executes when the condition is **false**.

Syntax: if (condition) {
//statements
}
else {
//statements
}

The code above will test the condition:
- If it evaluates to **true**, then the code inside the **if** statement will be executed.
- If it evaluates to **false**, then the code inside the **else** statement will be executed.

When only **one** statement is used inside the if/else, then the curly braces can be omitted.

Loops

A **loop** repeatedly executes a set of statements until a particular condition is satisfied.

A **while** loop statement repeatedly executes a target statement as long as a given condition remains **true**.

Syntax: while (condition) {
statement(s);
}

The loop iterates while the condition is **true**.
At the point when the condition becomes **false**, program control is shifted to the line that immediately follows the loop.

The switch Statement

The **switch** statement tests a variable against a list of values, which are called **cases**, to determine whether it is equal to any of them.**switch** (expression) {
case value1:
statement(s);
break;
case value2:
statement(s);
break;
...
case valueN:
statement(s);
break;
}
Switch evaluates the expression to determine whether it's equal to the value in the case statement. If a match is found, it executes the statements in that case.
A switch can contain any number of **case** statements, which are followed by the **value** in question and a **colon**.