1. Variable_DataType_TemplateString_Intellisense

```
export {}
 1
 2
     let message = 'Welcome back';
     console.log(message);
 3
 4
 5
     let x = 10;
     const y = 20;
 6
7
8
     let sum;
     const title = 'Codevolution';
9
10
     let isBeginner: boolean = true;
11
     let total: number = 0;
12
     let name: string = 'Vishwas';
13
     let sentence: string = `My name is ${name}
14
     I am a beginner in Typescript';
15
16
     console.log(sentence);
17
     total.
18
            ToExponential (method) Number.to

☆ toFixed

    ☆ toLocaleString

☆ toPrecision

            toString

    ∀ valueOf
```

2. Variable_Null_Undefined_Array_Tuple_Enum_Any_Multitype

```
let n: null = null;
19
     let u: undefined = undefined;
20
21
     let isNew: boolean = null;
22
23
     let myName: string = undefined;
24
     let list1: number[] = [1,2,3];
25
     let list2: Array<number> = [1,2,3];
26
27
28
     let person1: [string, number] = ['Chris', 22];
29
30
     enum Color {Red = 5, Green, Blue};
31
32
33
     let c: Color = Color.Green;
     console.log(c);
34
35
36
     let randomValue: any = 10;
     randomValue = true;
37
     randomValue = 'Vishwas';
38
39
     let multiType: number | boolean;
59
     multiType = 20;
60
     multiType = true;
61
```

3. Function Basic

```
66
      function add(num1: number, num2: number) {
67
                                                          Function with Parameters with
           return num1 + num2;
                                                          Data Type
68
      }
69
      add(5, 10);
      function add(num1: number, num2: number): number {
66
                                                                   Function with
67
           return num1 + num2;
                                                                   Parameters with Data Type
68
      }
                                                                   and Return Type
69
      add(5, 10);
      function add(num1: number, num2?: number): number {
66
67
        if (num2)
                                                                      Function with Optional
68
           return num1 + num2;
                                                                      Parameters with Data
69
         else
                                                                      Type and Return Type
70
           return num1;
71
      }
72
      add(5, 10);
      add(5);
73
                                                                   \mathbf{\Pi}
                                                                       function add(num1: number, num2: number = 10): number {
66
                                                                         Parameter Having
                                                                         Default Value,
67
        if (num2)
                                                                         another way of
68
           return num1 + num2;
                                                                         making optional
69
        else
                                                                         when calling
70
           return num1;
71
      add(5, 10);
72
      add(5);
73
```

4. Function WithObjectAndInterface

```
function fullName(person: {firstName: string, lastName: string}) {
74
         console.log(`${person.firstName} ${person.lastName}`);
75
     }
76
77
     let p = {
78
         firstName: 'Bruce',
79
80
         lastName: 'Wayne'
     };
81
82
     fullName(p);
83
     interface Person {
74
75
         firstName: string;
         lastName: string;
76
     }
77
78
     function fullName(person: Person) {
79
         console.log(`${person.firstName} ${person.lastName}`);
80
     }
81
82
     let p = {
83
         firstName: 'Bruce',
84
         lastName: 'Wayne'
85
86
     };
```

5. Class

```
class Employee {
90
          employeeName: string;
91
92
          constructor(name: string) {
93
94
               this.employeeName = name;
           }
95
96
          greet() {
97
               console.log(`Good Morning ${this.employeeName}`);
98
99
100
101
      let emp1 = new Employee('Vishwas');
102
      console.log(emp1.employeeName);
103
      emp1.greet();
104
                                                           2: powersl
PROBLEMS
          OUTPUT DEBUG CONSOLE
                                 TERMINAL
Bruce undefined
Vishwas
Good Morning Vishwas
```

6. Interface

```
class Manager extends Employee {
106
           constructor(managerName: string) {
107
               super(managerName);
108
109
110
           delegateWork() {
               console.log(`Manager delegating tasks`);
111
112
113
      }
114
      let m1 = new Manager('Bruce')
115
      m1.delegateWork();
116
      m1.greet();
117
118
      console.log(m1.employeeName);
PROBLEMS
           OUTPUT
                   DEBUG CONSOLE
                                  TERMINAL
Manager delegating tasks
Good Morning Bruce
Bruce
PS E:\Codevolution\Course\Typescript>
```