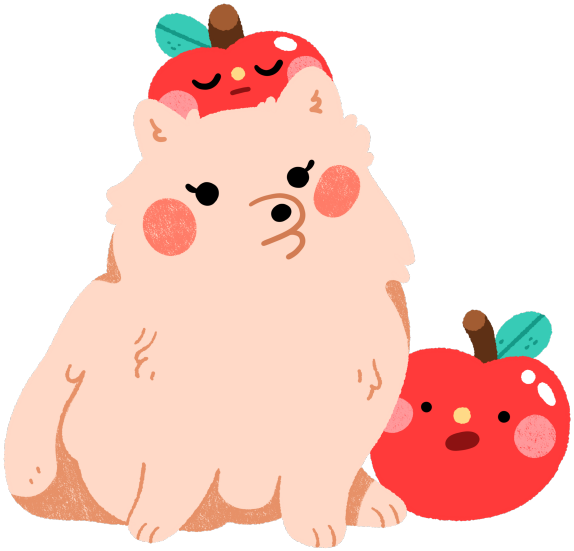

Starting at 7:05 am

Enjoy the song !!

Today's session

1. Celebrating Error
2. Data Types [float, double, String, boolean]
3. Typecasting [float vs double]
4. Typecasting [decimal vs non-decimal]
5. Taking input

ERROR

- your new soulmate

1. **How we feel about error**
    a. We are generally prejudiced about errors that they are a bad thing.
    b. They make us **disappointed** 😫 , **frustrated** 😕 and **angry** 😡
    c. **But this is not how we should feel when we get an error.**
2. **How should we feel about errors?**
    a. Getting errors is the best thing 🥳 that can happen to us when we are in our learning stage except **during our interviews** 😜
    b. **Why are errors the best thing?**
        i. Because research suggests that learning from mistakes can enhance your learning curve.
        ii. In order to know the right way of doing something, we also need to understand the wrong way of doing it and errors teach us exactly that.
        iii. That's why we need to celebrate our errors and make them the **love of our life even if you are in a committed relationship** 😂
        iv. So going forward whenever you get an error **jump in the air, raise your fist and say to yourself that "Yaaayyy!!! Now I get to learn something new".**
3. **Steps to approach any problem**
    a. Read the Problem Description **thoroughly** and understand it
    b. Read the Input and Output Format **[If any]**
    c. Read the Example Input, Output and Explanation **[If any]**
    d. Think of an idea/approach using Pen and Paper and solve the problem
        i. Check the correctness of the approach with an example
        ii. Check if it gives TLE or not **[Applicable only for DSA]**
    e. Write the code
        i. Things that can happen when we Run / Test / Submit code
            1. Code gets accepted 🤩
            2. You get Error 🥳
            3. Wrong Answer 😎

4. **What to do when we get an error?**
    a. Read the error message with line number and try to figure out yourself
    b. Take help from **Notes** or **Post-Lecture Content** to understand what went wrong
    c. Still not able to understand, **Google** the **Error message** and understand why this error occurs and try to resolve it
    d. After following all the above things, if you are still not able to resolve it, seek help from your **Friends/ TA/ Instructor** in the given order.

5. **What to do when we get a wrong answer?**
    a. Firstly, appreciate yourself for writing correct syntax 👍
    b. Only issue is the code output is not matching with the expected output.
    c. To rectify the issue follow the below steps:
        i. Compare your answer with expected output and find differences.
        ii. Take an example and dry run with **Pen and Paper** to understand what's going wrong.
        iii. If you are still not able to resolve it, seek help from your **Friends/ TA/ Instructor** in the given order.

Numbers

- Non decimal / Integers : int , long
- Decimal : float , double
  [numbers having decimal point]
  22.67 , 4.0 , -17.89 , etc.

1. By default any decimal number will be considered as double.
2. float x = 3.14; // gives error because 3.14 is double
3. float x = 3.14f; // now 3.14 is considered as float.
4. float can have upto 6-7 digits after decimal point
5. double can have upto 15-16 digits after decimal point.
6. decimal is more precise than float.

Typecasting Rules :

→ If there is no loss of data then no error : Implicit / Widening
→ If there is even slight possibility of data loss then error
We can still do it forcefully : Explicit / Narrowing

Taking Input :

```
Scanner scn = new Scanner (System.in);
float  → scn.nextFloat ();
double → scn.nextDouble ();
```

Text →

- String → whatever you write in double quotes is a String
- char → one single character/symbol

```
String s1 = "Hello123";
```

Taking input :

sc.next(); → only take input till space is encountered

sc.nextLine(); → takes whole line as input [untill enter is encountered]

Input: My name is Priyanshi

```
Scanner sc = new Scanner(System.in);

String str1 = sc.next();  // str1 → "My"
System.out.print(str1);   → My
String str2 = sc.next();  // str2 → "name"
System.out.print(str2);   → name
```

Output : Myname

Input: My name is Priyanshi
Hello Guys!

```
Scanner sc = new Scanner (System.in);

String str1 = sc.nextLine();
System.out.print(str1);
```
↳ My name is Priyanshi

```
String str2 = sc.next();
System.out.print(str2);
```
↳ Hello

Note: When we take input a String input using .nextLine() after a number / .next() [when the inputs are in different lines] we get empty string.

Input: 46
Hello World

```
Scanner sc = new Scanner (System.in);
int n = sc.nextInt();          // n = 46
String s1 = sc.nextLine();     // s1 = "" [Empty string]
String s2 = sc.nextLine();     // s2 = "Hello World"
```

Input: Hey
Welcome Everyone

```
Scanner sc = new Scanner (System.in);
String a = sc.next();        // a = "Hey"
String b = sc.nextLine();    // b = " "  [Empty string]
String c = sc.nextLine();    // c = " Welcome Everyone"
```

boolean [true / false]

```
boolean x = true;
S.O.Pln (x);  ⟶  true
```

Taking input:

```
boolean y = sc.nextBoolean();
```