Array:

String[] array\_name; //data\_type arr\_name[arr\_size]; in C

array\_name = new String[array\_size];

forEach:

for(int i:collection\_name){

System.out.println(i)

}

1. Here i represents the items of the collection and Note: its not a index like in normal loop.
2. Only i will be used to in println() statement not a[i].
3. Here i is a Counter Variable not Index.

**Package is JAVA**

* Package is a collection of classes.(which contains at least two packages)
* There are two types of Packages we can use in Java:

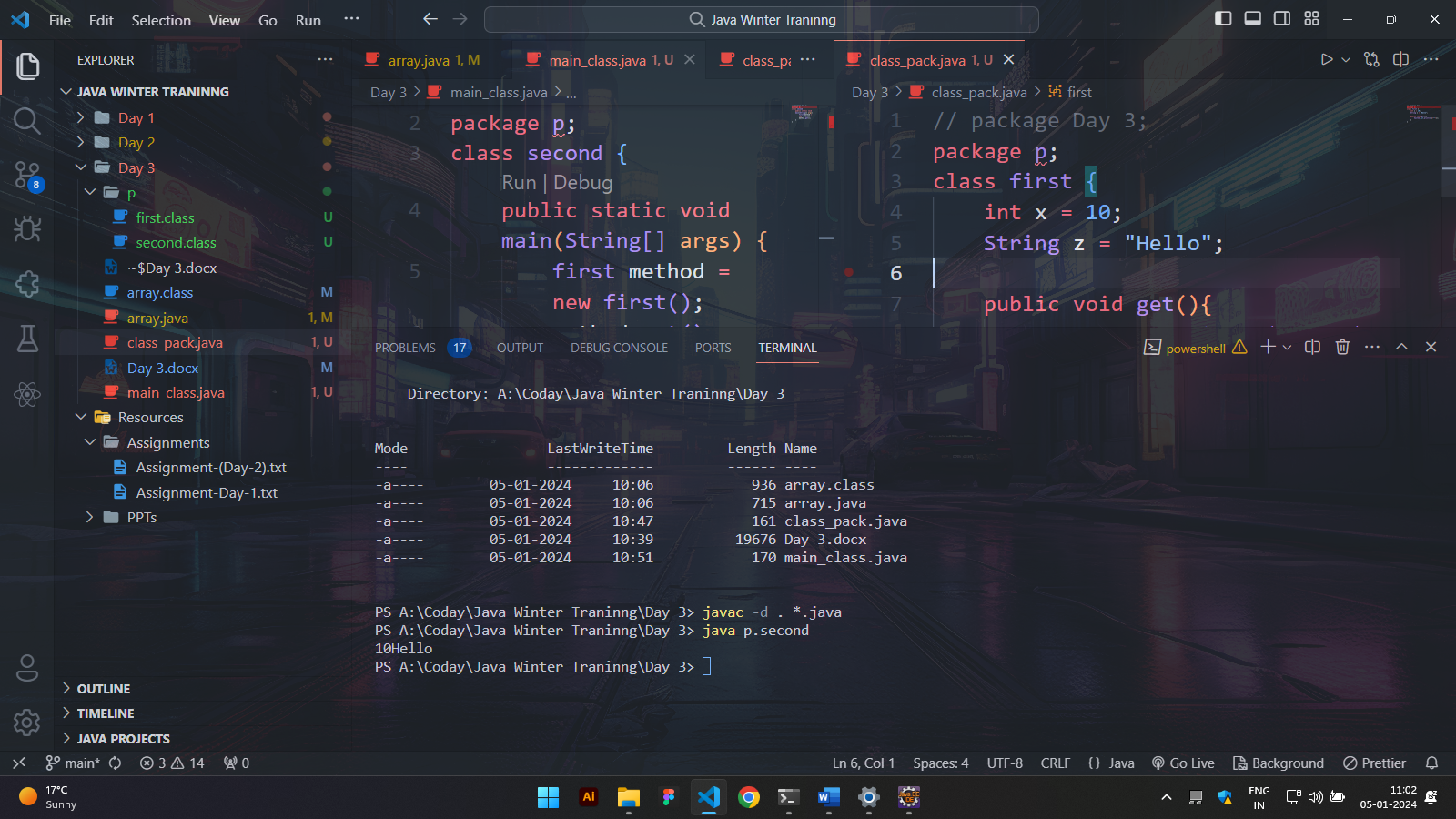
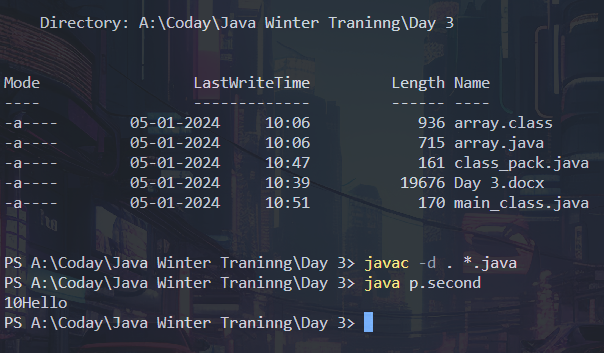
1. Built-in Packages (Pre-defined packages) e.g. 🡪 lang,util,io etc

* Lang
* Thread
* Util
* io

1. User-Defined Packages

**How to Create User defined Packages in Java?**

* We can use method of different class in different class.
* javac -d . \*.java // This command makes package example can be seen on this page.



import will be second and package will be first

**Access Modifiers**

**Types**

* default
* public
* protected
* private

will it Run Table?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Access Modifiers | default | public | protected | private |
| Same class | Yes | Yes | Yes | yes |
| Class in same package | Yes | Yes | Yes | No |
| Subclass in same package | Yes | Yes | Yes | No |
| Subclass outside same package | No | Yes | Yes | No |
|  |  |  |  |  |
|  |  |  |  |  |

* By default if modifier not defined then it is set to be default
* We can save java file with any name but during run we have to call the class in which main method exists.
* The class in which main method exist , could not be private.
* If both public then save with main method class name.

Sub-Classes in Java

* Parent Class/Super class/Base class
* Child Class/Derive class (main method)

Inheritance in JAVA

* A mechanism to access the traits of parent in class by child class.
* extends keyword is used.
* Types:-
  1. Single inheritance // Imp
  2. Multiple inheritance //Imp
  3. Multilevel inheritance
  4. Hybrid inheritance

Single inheritance:

Class 1

Abstraction

* b important features of OOPs Paradigm which hides the implementation and only shows the application.
* Two Way:-
  1. Through Abstract class: gives probability so may we can achieve to some extent or not.
     + abstract keyword is used.
  2. Through interface(100%).

Syntax:-

abstract class class\_name{

methods(abstract or non-abstract)

abstract void get(); //abstract method – No Body

abstract void get(){

//Non-abstract method – As Body and we shown information

}

}

interface interface\_name{

//methods(abstract)

void get(); //abstract method and no abstract keyword is required as by default it is.

//Cant define a Non-abstract method inside a interface

}

* From Security pov, Interface is Better to attain abstraction.
* Class 🡪 class = extends Keyword
* Interface 🡪 Class = implements Keyword
* Interface 🡪 Interface : implements Keyword
* In Derived Class the Access modifier of method will be stronger than the method declared
* super(); is a keyword used to refer parent class variables,methods and constructors.