Day 1

1.
Accept i/ps from User , till user enters "quit" or any other option.
I/P : operation(add|sub|mult|div) , number1(double) , number2(double)
Display the result of the operation.

2. Accept 2 double values from User (using Scanner). Check data type.
. If arguments are not doubles , supply suitable error message & terminate.
If numbers are double values , print its average.

3. Display food menu to user. User will select items from menu along with the quantity. (eg 1. Dosa 2. Samosa ....10 .
 Generate Bill) When user enters 'Generate Bill' option, display total bill & exit.

Day 2
Practice lab
 Please go through ready made code samples --Test1,Test2,Test3,Test4 & guess the o/p
(can exclude bit wise operators currently)
 Please go through ready made code samples --on data types.

1. Solve Tank assignment along with memory picture.

```
class Tank {
  private int level;
  Tank(int l)
   {
     level=l;
   }
  public void setLevel(int level)
  {
    this.level=level;
  }
  public int getLevel()
  {
    return level;
  }
}

public class Assignment {
  public static void main(String[] args) {
    Tank t1 = new Tank(10);
    Tank t2 = new Tank(20);

    s.o.p("1: t1.level: " + t1.getLevel() +
          ", t2.level: " + t2.getLevel());
    t1 = t2;
    s.o.p("2: t1.level: " + t1.getLevel() +
          ", t2.level: " + t2.getLevel());
    t1.setLevel(27);
     s.o.p("3: t1.level: " + t1.getLevel() +
          ", t2.level: " + t2.getLevel());
```

```
    t2.setLevel(t1.getLevel()+10);
     s.o.p("4: t1.level: " + t1.getLevel() +
          ", t2.level: " + t2.getLevel());
  }
}
```

2.
Create Point class Point2D , under package com.app.geometry : for representing a point in x-y co-ordinate system.

1.1 Create a parameterized constructor to accept x & y co-ords.

1.2 Add public String show()) --to return point's x & y co-ords

1.2 Add isEqual method to Point2D class : boolean returning method : must ret. true if both points are having same x,y co-ords or false otherwise.
:
(boolean isEqual(Point2D p)

1.3 Add a method to Point2D class -- to create and return new point having given x & y offset.
(eg Point2D createNewPoint(int xOffset,int yOffset))

1.4 Add calculateDistance method to calculate distance between current point & specified point & return the distance to the caller.
(eg double calcDistance(Point2D p2))
Hint : Use distance formula . Use java.lang.Math class methods --sqrt, pow etc.

1.5 Write TestPoint class , under "tester" package with a main method
Accept co ordinates of 2 points from user (Scanner) --p1 & p2

1.6 Use show method to display point details.(p1's details & p2's details)

1.7 Invoke isEqual & display if points are same or different (i.e p1 & p2 are located at the same position)

1.8 Create new point p3 , with the dimensions offset from p1.
I/P --x offset & y offset

1.8 Display distance between 2 points.(between p1 & p2)

----------Corona Day 3----------

Day 3
Create new eclipse project day3_assgn
Copy earlier Point2D class
Create a TestPoints class in "com.app.tester" package for the following.
1. Prompt user , how many points to plot.
Create suitable array.
2. Add a menu , Run the application till exit.
1 Plot a new point
I/P --index , x & y
2 Display all points plotted so far.
3. Calculate distance

I/P strt , end points
10. Exit

==========
==========
Day 4

1. Create Person,Student & faculty classes(under com.app.cdac package as discussed. Add suitable parameterized constructors & override toString

1.1 Add new method in Student class
public void feedback() {
  //display the message --student's first name submitted feedback
eg : Rama submitted feedback
}

1.2 Add new method in Faculty class
public void evaluate() {
  //display the message --faculty's last name evaluated students
eg : Kher evaluated students
}

1.3 Create a EventManager class in "tester" package with main method.

1.4 Create suitable array to store participant details.

1.5 Add Options
1 : Register New Student
2 : Register New Faculty
3 : Display all participant details (using for-each & toString)
4 : Find Details of a specific participant
i/p -- seat no.
o/p -- specific participant details(toString)
5. Invoke specific functionality
i/p --seat no
o/p -- in case of student --invoke it's feedback method or in case of Faculty --invoke evaluate method.
10 : Exit

Day 4.2

2. Fresh business scenario to apply inheritance , polymorphism (upcasting/downcasting) to emp organization scenario.

Create Emp based organization structure --- Emp , Mgr , Worker
All of above classes in --com.app.org

Emp state--- id(int), name, deptId , basic(double)
Behaviour --- get emp details -- via toString
compute net salary ---ret 0
(eg : public double computeNetSalary(){return 0;})

Mgr state ---id,name,basic,deptId,basic , perfBonus
Behaviour ----1. get mgr details(toString)
2. compute net salary (formula: basic+perfBonus)
3. get performance bonus.

Worker state --id,name,basic,deptId,hoursWorked,hourlyRate
Behaviour---
1. get worker details(toString)
2. compute net salary (formula: = basic+(hrs*rate)
3. get hrlyRate of the worker

Organize classes in inheritance hierarchy.

Tester

Write TestOrg in "tester" package.
Create suitable array to store organization details.
Provide following options

1. Hire Manager
2. Hire Worker
3. Display information of all employees , net salary & perf bonus of mgr or hourly rate of the worker using single for-each loop.
10 Exit
==========
Day 5
1. Complete bank scenario & test it.

2. Apply interfaces.

1.Create Computable interface.
Declare functionality to calculate area & perimeter of a bounded shape.
Add PI as a constant.

Develop follwoing classes

2. Point --- x,y , constructor , toString

3. Circle --x,y,radius,constructor , toString, calc area & perimeter +
public void drawArc(int angleInDeg)
{ sop("drawing arc with angle in radian"); }
ang in rad=(ang in deg*180)/PI

4. Rectangle --x,y,width,height,constructor , toString, calc area & perimeter +
public void diagonals()
{
 sop("diagonals : "+value of diagonal);
}

5. Tester --Add a main method along with Scanner.
5.1 Prompt user for how many bounded shapes. Create suitable array.
Options
1. Add Circle
2. Add Rectangle
3. Display area & perimeter of all shapes, using for-each loop.
In the same for-each , invoke drawArc for a circle or diagonals for a rectangle type.

==========
==========
Day 6
Create java application for bank handling scenario along with validations.

1. BankAccount -- acct no(int),customer name(string), acct type(string),balance(double)
toString , constr.
Methods
public void withdraw(double amt)
public void deposit(double amt)
public void transferFunds(BankAccount dest,double amt)

2. SavingAccount -- acct no,customer name, acct type,balance,savingIntRate
Add a method
public void applyInterest()
--to apply simple interest (yearly)

3.CurrentAccount -- acct no,customer name, acct type,balance,daily withdrawl limit.
public void modifyLimit(double limit)

4. Validation rules(add it in a separate class AccountValidationRules & add static method for validation)

Min balance for ANY type of a/c =1000
Later add below rules.
Account types supported -- saving , current
Customer name --min len 4 max length 10

a/c transaction date must be currnt financial year

5. Create a Tester1 with Scanner (try-with-resources)

Accept a/c details from user, validate the i/ps
In absence of validation errs , create suitable a/c & display its summary(via toString)
In case of errs --show err mesg , via catch block.

==========
==========
Day 7
Create Tester 2 with scanner , complete with menu .
Create suitable array to store bank account details.
1. Open saving account
I/P acc no,customer name, ac type , balance , interest rate
2. Open current account
I/P acc no,customer name, ac type , balance , daily limit
3. Show a/c summary
I/P acc no

4. Withdraw
I/P acc no , amount
5. Deposit
I/P acc no , amount

6. Transfer Funds
I/Ps src acc no, dest acc no , amount

100. Exit

==========completed==========

Day 7
1.Revise string, date/time handling examples.

2. Solve this
Create a Vehicle class , having chasis no(string),price(double),date of manufacturing(java.util.Date)
Add suitable constructor & toString
Add suitable method of cheking equality of 2 vehicles
equality criteria -- 2 vehicles are same iff chasis no & date of manufacturing is same.

Create TestEquals class.
Accept 2 vehicle's details from User & create Vehicle objects.
Display "SAME" or "Different" suitably.(by testing equality)

==========day 8==========

Day 8
1. Create Emp class having
id(int) , deptId(string), name(string), salary, joinDate(Date)
Add static init block ,constr , toString , equals suitably.
Emp's identity -- 2 emps are same iff id & dept id is same.

Create RecruitEmps as Tester with main(...)
Accept 5 emp details in a for loop & display their details using toString.
Validation rule --Can't hire duplicate emps.
---
2. Apply enum
Replace string dept id from Emp class by enum.
Modify constr , toString , equals suitably.
Add a validation rule --
While recruiting --emp must be assigned to specific depts only.
Otherwise , raise custom exception.
----------day 8.2----------
3. Create an enum --CustomerTypes
SILVER,GOLD,PLATUNUM

Create a customer class
state --name,email,password,dateOfBirth,custType(enum),reg amount(double)
Equality testing (Primary Key) -- email (i.e 2 customers are same iff email is same)
Validation rules
1. Don't register customers with duplicate email ids.
2. customer type must be either SILVER or GOLD or PLATUNUM
3. customer's age must be > 21

Tester
Supply following options
1. Register new customer

2. Display all customer details
3. Display customer details of a specific type
(type will be given by user)
==========
Day 9
1. Create an outer class Emp -- id,name,salary, with suitable constrcutor & toString.
Emp HAS-A Address
Business logic rule --User should be unable to create address instance w/o Emp class instance.
Address -- city,state,country,phoneNo --string

Create a Tester , with a scanner.
Accept 1 emp details (w/o address) & create the instance.
Accept address details & assign address to an emp.
Display details --to show emp n address info.

2. Create a class Holder , that can hold ANY type of the data.
(i.e any primitive or any ref type)
Supply constr & a getter.
Create a tester with main(no scanner)
Create a Holder class instance holding int data
(Holder HAS-A integer)
Create another Holder class instance holding string data
(Holder HAS-A string)

3. Create ArrayList of integers & populate it using scanner
Check if it allows dups / maintains order / sorted or un sorted.

3.2 Create an int[] with some ints.
Use this to populate AL.
1. check if a particular element exists in the list.
2. Display elements of the list
3. disp index of 1st occurance of the elem
4. double values in the list --if elem val > 20
5. remove elem at the specified index

4. Apply understanding of AL to practical scenario
4.1 Copy earlier Emp & Address class in to new project.
4.2 Create a Tester for the following options
1. Hire Emp
I/P --emp id name , salary
(no address)

2. Display all emp's details ,with individual emp detail on seperate line.

3. Display if specific emp exists or not.
I/P --emp id
O/P --Print emp details if found or throw custom exc , to show err mesg.

4. Assign emp address
I/P emp id & address details (city,state,country,phone no)
O/P --Print success mesg if emp id found with linked address , or throw custom exc , to show err mesg.

5. Update Emp's salary.

I/P emp id & salary increment.
O/P --Print success mesg : salary updated or throw custom exc , to show err mesg.

6. Fire employee
I/p emp id
O/P : Print success mesg : emp details removed or throw custom exc , to show err mesg

7. Display all emp names earning salary > specified salary
I/P : threashold salary.
O/P Emp names.
---
8. Practice lab work :
Add 2 data members in Emp class
private Date joinDate;
private Department dept;//Department -- enum
Make necessary mofications, in Emp's constr , toString
Add methods in ValidationRules
1. dept name --RND,HR,MKTG,SALES
2. join date must be in current financial year.

9. Display all emp names from a specified dept , joined after a specific date.
I/P : dept name & join date.
O/P Emp names.

10. Sort the emps in asc order as per emp id

11. Sort the emps in asc order as per emp salary

12. Using custom ordering , sort emps as per dept id & join date

Tips
1. How to find user's age
Look at API of LocalDate --now,of
Look at API of Period --between & getYears.

Reading H.W
Go through --non static nested class , generic class syntax & the need of it. Revise collection framework overview , till ArrayList from slides & its readme.

==========

Day 10
1.Create String[] & populate it .
Populate ArrayList<String> from it.
Attach iterator to display the strings from AL.
Remove all the strings with len > 3

2.Set up n understand , Iterator/ListIterator related exceptions.

3. Use ListIterator to display list contents in reverse manner.

4. Create CollectionUtils class , in utils package.
Add a static method to ret populated ArrayList with some emps.
==========

Day 11
1. Create User class -- id(int),email(string),name(string)
Add suitable constr, to String.
2 users are same iff -- id & email is same.
Add at least 5 users in a HashSet & confirm your understanding of hashing algo.
Add 1 st user details.
Then :
Test cases -- same id, diff. email
Test cases -- same email , diff id
Test cases -- same id, same email
Test cases -- diff id , diff. email

2. Create a LinkedHashSet of integers & confirm if its un sorted BUT ordered.

3.
User details must be stored in a suitable data structure.
Ensure --Duplicate user details are not getting stored & user details are displayed as per descending reg amounts.

==========
for day 12 and 13 day 9th assignment extended

==========collection and File i/o==========
Day 14
Apply Collection n I/O , along with Java 8 API.
1.1 Create java application , StoreData
Use Collection Utils , to get populated AL of students.
1.2 Sort it as per student's GPA & store it text file in buffered manner.

2. Create another java application , RestoreData
2.1 Restore details from text file .(restore it in any suitable collection).
Add 2 more student's details .
Display name of the overall topper.
(Try to use Java 8 streams with lambda expressions as much as possible).

==========
Day 15
1. Create java application for customer management .

1.1 Customer class

data members -- email(unique),password , reg amount,regDate(LocalDate),
type (enum : SILVER/GOLD/PLATINUM)
address(home address , office address,hostel address etc.)
Provide suitable constr to accept all details excluding address.
(Customer HAS-A Address)
How will you add multiple addresses?
Address -- city,state,country,phoneNo,type(home/office/hostel...)
Add a method linkAddress, to assign address to customer
eg : In Customer class ,
public void linkAddress(String city,String state,String country,String
phoneNo,String type)

1.3 Create a Tester for following options.

Restore customer details from bin file using de-serialization.

1. Register new customer
I/P --customer details
Should not register duplicate customer(with the same email)

2. Customer Login
I/P : customer email , password
O/P : mesg (login successful / invalid email / invalid pwd)

3. Link Address(Try last!)
I/P : customer email , password , adr details

100. Exit
Before terminating from application store customer details in bin file
using serialization.
==========
==========
Day 16
No new assignment.
Pls complete pending work.

==========
==========
Day 17
1. Create multi threaded java application --- having 3 threads
main , even , odd
Create EvenPrinterTask & OddPrinterTask
Accept from user (in main thread)-- start & end values.
Even no printer should print even nos within range, with small sleep.
Odd no printer should print odd nos within range , with small sleep.

Solve above using implements Runnable scenarion. Ensure no orphans.

2.Create multi threaded java application for the following(MUST use
implements Runnable approach)
Student class -- rollNo(int),name(string),marks(int),dob(LocalDate)
2.1 main thread -- get a populated(hard coded) HM<Integer,Student>
Pass the same HashMap to child threads.

2.2 1st thread should sort student details as per marks & save it in a text file.
2.3 2nd thread should sort student details as per dob & save it in a separate text file.
3. Create a Joint Account scenario & test it with 2 threads, representing 2 customers operating upon same a/c.
1. Create Account class with balance as state.
Add a constr.
Add 2 methods
1.1 public void updateAccount(double amt)
{
   //update balance
  //think time
  //cancel updates
}
1.2
Check Balance
public double checkBalance()
{
  //return a/c balance. --add some sleep to simulate dly in fetching balance.
}

2. Create Updater task , which will invoke update balance method , continuously.

3. Create BalanceChecker task which will invoke check balance method , & terminate JVM (System.exit(1)) , in case of wrong balance.

4. Create Tester
Create joint a/c , with some init balance. Share it with 2 thrds , start them. Ensure no orphans.
==========
==========
Day 18
1. Applying thread safety to thread un-safe collections.
1.1 Create an empty ArrayList<Integer> with large initial capacity(20,000)
Populate 1000 refs.
1.2 Using anonymous inner class or lambda expression, create 2 threads
Share the SAME List between , Reader & Writer threads.
1.3 A Writer thread , should add more 10000 refs to the ArrayList.

1.4 A Reader thread should read the list data using Iterator/ListIterator or for-each.

Observe n conclude.

2. Modify Day17's student data sorting assignment to write sort by date & sort by marks information to the SAME data file
(Hint : use auto flush option of PrintWriter to first check the problem. Then apply synchronization as the solution)

3. MUST install MySQL database server & run Test DB connection java code to confirm DB connectivity.

==========
==========

Day 19
1. Display max salary of the employee from specified dept.
2. Insert new emp details
I/P name, adr,sal,dept join date
O/P -- message
sql="insert into my_emp values(default,?,?,?,?,?)";
3. Update emp details
I/P -- emp id,new dept,sal incr
O/P -- message
sql="update my_emp set deptid=? ,salary=salary+? where empid=?";
4. Delete emp details
I/P emp id
o/p message
sql="delete from my_emp where empid=?";
5. Show depatment summary , consisting of
dept name , no of emps(count),avg salary(avg),max salary,min salary,sum of salary
for all the deparments in your organization.
sql="select
deptid,count(*),avg(salary),max(salary),min(salary),sum(salary) from my_emp group by dept";

----------case 2----------

Create a layered standalone JDBC application for the following
(Layers Tester -- DAO (uses DBUtils) -- POJO/Entity --DB)
ref : my_customers table.

1. Customer Login
I/P --email,password
O/P Success mesg with customer details or err mesg via custom exception
2. Customer registration
I/P reg amount ,email , name ,pwd ,reg date, role
(validation : shouldn't accept dup email)
O/P message
3. Apply discount in reg amount , to all customers registered before a specific date.--allowed only for admin role.
I/P credentials (email,password) discount amount , date
O/P Display how many customers were offered discount or error message -- You are not authorized.
4. Un subscribe customer -- allowed only for admin role.
I/P credentials (admin's email,password) , customer email to delete.
O/P Display customer un subscribed or error message --You are not authorized / or customer doesn't exist.

5. Top N analysis --allowed only under admin login.
Display 2 customer(not admin!) names paying maximum reg amount

6. Optional (Can try this after understanding joins/Foreign keys in DBT module)
Create a products tables with
id (PK) , name (unique) , price , exp_date , category
Populate it with some products.
Create orders table with
order_id(PK) , customer id (FK references customer's id) , product_id(FK references product's id),order date , quantity , amount

6.1 After successful customer login , display products in the shop.
Option : Place Order
I/P product name , quantity.
It should add a record in orders table.
O/p A message : Order placed for a customer name with total order amount = ...

==========

Day 20
1. Execute a stored procedure for funds transfer.
2. Complete earlier pending CRUD work in JDBC(day 19 assignment first)
3. Execute a stored function for displaying customer level(silver/gold/platinum)
I/P cusotmer id
O/P Error mesg : invalid customer id or customer level
4. If time permits , launch TCP server & client on 2 different machines & test connection.

Day 21 optional

Optional assignment

Create java application for the following.
Create Student class in Student.java
2.1 Every student must have --- id(int) --if possible system generated(Hint : static data member), name (String), email(String), age(int),gpa

2.2 Add suitable parameterized constructor.
Accept -- name,email,age only from user.

2.3 Add a method fetchDetails to fetch student details
(ret type -- String)

2.4 Add computeGPA method in Student class
Accept 3 scores for quiz , test & assignments
gpa should be computed on 20 % of quiz score, 50% of test score & 30% of assignment score.

2.5 Write a TestStudent class in TestStudent.java

Accept 2 student details & display the same.
Compute GPAs by passing scores.
Display name of student having higher GPA.