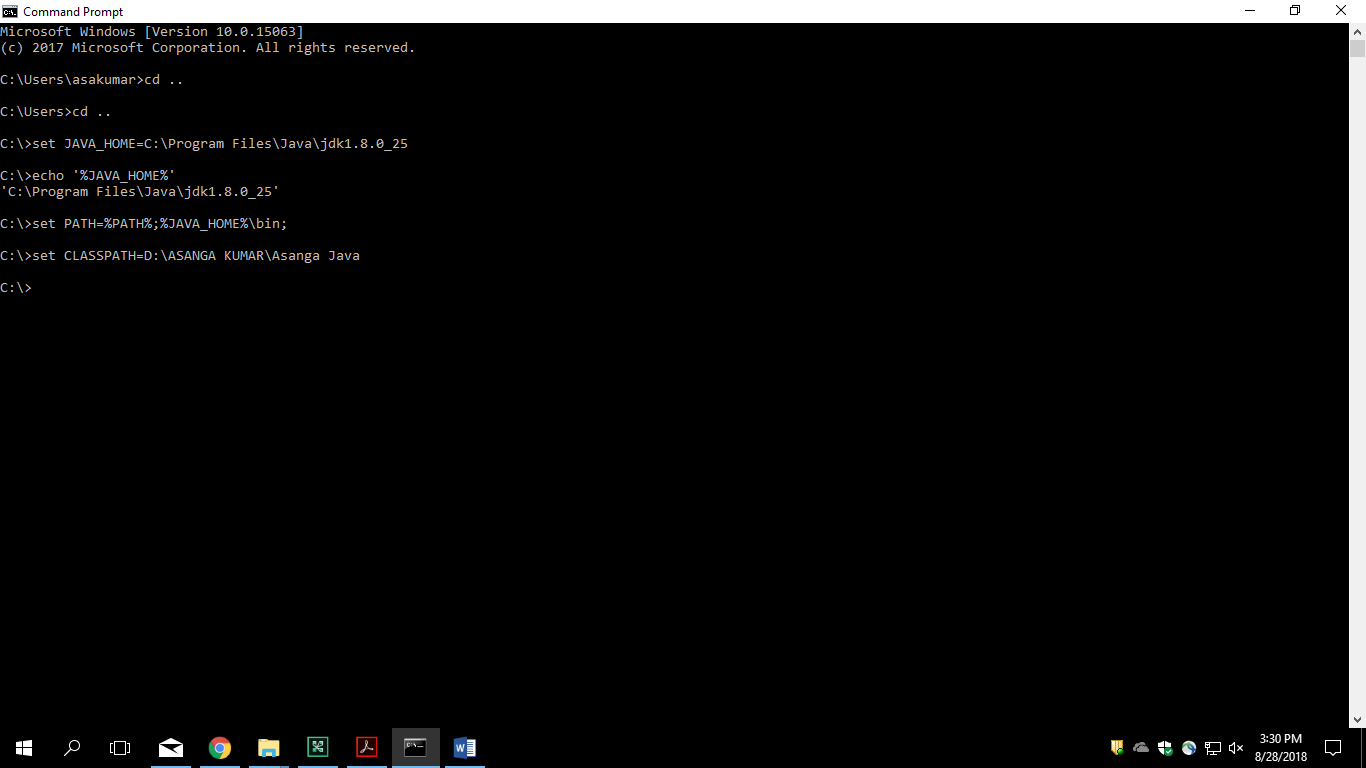
**Lab 1: Working with Java and Eclipse IDE**

1.1: Setting environment variables from CommandLineSolution:



**Lab 2: Getting Started**

2.1 Write a java program to print person details in the format as shown below:

Person Details:

\_\_\_\_\_\_\_\_\_\_\_\_

First Name: Asanga

Last Name: Kumar

Gender: F

Age: 20

Weight: 85.55

/\* package whatever; // don't place package name! \*/

import java.util.\*;

import java.lang.\*;

import java.io.\*;

/\* Name of the class has to be "Main" only if the class is public. \*/

class Ideone

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

System.out.println("First Name:Asanga \n Last Name: Kumar \n Gende: F \n Age: 20 \n Weight: 85.5");

}

}

**Lab 3: Basic Language Constructs**

3.1: Write a program to accept a number from user as a command line argument and check whether the given number is positive or negative number.

/\* package whatever; // don't place package name! \*/

import java.util.\*;

import java.lang.\*;

import java.io.\*;

/\* Name of the class has to be "Main" only if the class is public. \*/

class Ideone

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

Scanner in = new Scanner(System.in);

int s= in.nextInt();

if (s>0)

{

System.out.println("The no is positive");

}

else

{

System.out.println("The no is negative");

}

}

}

**Lab 4: Classes and Objects**

4.1 : Refer the class diagram given below and create a person class.

Create default and parameterized constructor for Person class.

/\* package whatever; // don't place package name! \*/

import java.util.\*;

import java.lang.\*;

import java.io.\*;

/\* Name of the class has to be "Main" only if the class is public. \*/

class Person

{

public static void main (String[] args) throws java.lang.Exception

{

// your code goes here

}

Person()

{

System.out.println("Hello");

}

Person(String name)

{

System.out.println("Hello"+ name);

}

}

**Lab 5: Extending Classes**

5.1 : Create Account Class as shown below in class diagram. Ensure minimum balance of INR 500 in a bank account is available.

/\* package whatever; // don't place package name! \*/

import java.util.\*;

import java.lang.\*;

import java.io.\*;

/\* Name of the class has to be "Main" only if the class is public. \*/

class Person

{

public String name;

public float age;

public Person()

{

}

public Person(String name, float age)

{

this.name=name;

this.age=age;

}

public void setname(String name)

{

this.name=name;

}

public String getname()

{

return this.name;

}

public void setage(float age)

{

this.age=age;

}

public float getage()

{

return this.age;

}

}

class Account

{

long accNum;

static long num=0;

double balance;

Person accHolder;

Account()

{

}

Account(String name, float age, double balance)

{

Person obj=new Person(name,age);

this.accHolder=obj;

this.balance=balance;

this.accNum=num+1;

num+=1;

}

void withdraw(double bal)

{

if((this.balance-bal>500))

{

this.balance-=bal;

}

}

void deposit(double bal)

{

this.balance+=balance;

}

double getBalance()

{

return this.balance;

}

public static void main(String args [])

{

Account ac1= new Account("Smith", 20,2000);

Account ac2= new Account("Kathy",21,3000);

ac1.deposit(2000);

ac1.withdraw(2000);

System.out.println("Smith balance: "+ ac1.balance);

System.out.println("Kathy balance: "+ ac2.balance);

String one= ac1.toString();

System.out.println(one);

}

public String toString()

{

return ("accholder: "+ this.accHolder.name+"balance: "+this.balance);

}

}

5.2: Extend the functionality through Inheritance and polymorphism (Maintenance)

Inherit two classes Savings Account and Current Account from account class. Implement the following in the respective classes.

a) Savings Account

a. Add a variable called minimum Balance and assign final modifier.

b. Override method called withdraw (This method should check for minimum balance and allow withdraw to happen)

b) Current Account

a. Add a variable called overdraft Limit

b. Override method called withdraw (checks whether overdraft limit is reached and returns a boolean value accordingly)

class Account {

  double balance;

  boolean withdraw(double bal){

    this.balance = this.balance - bal;

    return true;

  }

}

class SavingAccount extends Account{

     private final double minBalance=500;

    public SavingAccount(double bal){

        balance=bal;

        }

    public boolean withdraw(double bal){

        if((this.balance-bal)< minBalance)

            return false;

            else

             return true;

}

}

class CurrentAccount extends Account{

    private double overdraftLimit=-2000;

    public CurrentAccount(double bal){

      balance=bal;

    }

    public boolean withdraw (double bal){

        if((this.balance-bal)< overdraftLimit)

            return false;

            else

            return true;

    }

}