9-8-2024

Developer, Tester,ProjectManager

Repository – Project

Tuple – Relation

Merged with master --- Continuous Integration -- gitlab runner,Jenkins

Git clone -u <url> -

Drive –

Rootfolder /etc,/dev

Git - Version Management – Source code management

Gitlab – collaberative development platform / source code management server

To add files to the staging area- git add <filename>

Eg:git add sample.txt

Push changes from local repository to remote repository

Git push <url> <branch>

To understand current locally

Git branch-

Alias name – short name assigned to remote repo url

Git remote add <aliasname> <url>

Eg:git remote add Pankaj <http://gilab.stackroute.in/anil/actalent.git>

git pull -- to pull the changes from remote repo to local repo

git pull <url> <branch>

eg:git pull anil master

to create a new branch locally and change to that

git checkout -b developer1

git init—command to convert an existing folder to git folder

agile -

scrum -- , Kanban,

Product backlog->

CustomerManagement->sprints-> sprint team->sprint backlog->

Order Management->

Scrum Master ->

sprints->

10-8-2024

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

flow chart,psedocode,

algorithm

start

step1:

stop

graphical representation of algorithm

psedocode -

Programming --- procedure oriented – cobol, basic

Function oriented - c,

object oriented, component oriented

class stock{

int stockCode;

void AddStock(){

int a;

}

Void ShowStock(){}

} //encapsulation

Int main(){

Customer ptr = new Customer(); --- ptr is implicit pointer

Ptr.RegisterCustomer();

stock s = new stokc();

} -- stand alone functions

class Customer {

char name[100];

Void RegisterCustomer(){}

}

Sample.c ->compilation-obj file – machine code-> link+make ->exe file

Obj 100kb- exe -200mb --

St – memory management- pointers to

Encapsulation, abstraction,

Platform independent –

Java runtime environment –

.jar -

12-8-2024

OOA –object oriented analyses --

Identity the data

Identity the process

Identity related data and processs -- consider it as an object

Object is implemented by – class,struct,record,enum,union etc

Stand alone functions -- functions that does not use the global data of any object

Java – pure object oriented -- object/class definitions

Stand alone functions in java are called as static

Make, link – combine object files to a single deployable unit . exe

-jar - runtime --

Java – packages – collection of related classes

To create package

Javac -d <destination> <source>

Eg : javac -d . \*.java

To create java documentation

Javadoc -d <destination> <source>

Jdk – PrintStream out

System.out.println();

Building b = new Building();

Building b = 10; - strongly typed language -- truncation of data

String name = new String(“anil”);

String name = “anil”; -- boxing --new String(“anil”); --- auto boxing

Int x=50;

Integer a =20;

System.out.println(name); -unboxing – auto unboxing

float rate = (float)20.5;

variable declaration statement

assignment statement --- syntax

arithmentic statement

decicision making statement

function calling

loop statements

relational statements

input statement

output statement

etc

in java all statements are written inside a function and declaration statements can be inside class also

primitive -- extended/userdefined/complex/composite

byte,short,int,long,float,double,Boolean,char - u

int a; --- value types

String s; --- reference -- 4 byte,8 byte

Operators - arithmethic,relational,logical operators,arithmetic assignment,bitwise operator ,ternary

1010

0010

1000

Loop – exit controlled,

Int count =0;

Do{

System.out.println(count);

Count++;

}while(count<10);

While(){}

For(int count=0;count<10;c++){

}

For(int x in [1,2,3,4]){

X=10;

}

Write a program to display multiplication table of number 5

Arrays - group of variable

Single dimensional, multidimenstiona

If(s==s1)

Int [][]arr[] = new int[2][2][2] ;

Int[][]marks=new int[10][5];

Arrays reference int []xx = null;

findAverage(null);

call by reference and value

Array or class type call reference

Void FindAverage(int []input){

}

FindAverage(null);

Array –

* RealEstate domain -
* Property – Flat,House,Land,Plantation,Office,CommercialSpace etc
* Area,saleType,Location,Rate
* Flat – floor,deposit
* House – bhk,isparking,deposit
* Office –
* Commercial space –
* Relationship – kind of, has a, uses
* Inheritance, composition ,utilization
* DRY – do not repeat yourself
* SOLID –
* Class Driver{
* Public void Driving(){
* C.switchon();
* C.gearchange();
* C.accleration();
* }
* Public void eating(){}
* }
* TS 01 AA 1001 – is a maruthi alto car
* Class servicestation{
* Doservice(Vehicle c){}
* }
* Class vehicle{}
* Class car extends vehicle{}

Servicestation().Doservice(new Car());

Property f = new Flat();

f.accept(); --- binding --- associating function to a class

compile time -- static binding --compiler

runtime --- dynamic binding -- runtime

abstract methods – if used bottom up approach it used for dynamic binding

top down approach – the abstract method will act as a contract

www ,ieee –

interfaces -- to create specification of kind of classes,

interface Vehicl{ void acceleration();void braking();void gearchange();}

UML—JUDE

class BMW implements Vehicle {

}

Interfaces will to implement dynamic binding –

Interfaces are used future referencing –

Interfaces can be used higher level of abstraction --

Abstraction – provide relevant information from users perspective

Same class method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Access specificer | sameclass | Anotherclass in same package | Child class in samepackae | Another class in another package | Child class in another package |
| public | Yes | yes | Yes | yes | yes |
| friendly | yes | yes | yes | no |  |
| protected | yes | Yes | yes | no |  |
| private | yes | no | no | no | No |

another class ,another class method in the same package

another class or method in another package

class Atm implements Customer,Employee{

private Atm(){}

public void checkBalance(){}

public void setDenomination(){}

public static Atm getInstance(){

return new Atm();

}

}

Public interface Employee{ public void setDenomination();}

Public interface Customer{ public void checkBalance();}

Employee e = Atm.getIntance();//new Atm();

Customer c = Atm.getInstance();//new Atm();

e.setDenomination();

c.checkBalnace();

component- beans

1. Introspection --
2. Customization
3. Interaction -

JFrame jf=new JFrame();

Jf.setLocation(12,12);

Jf.setBounds(12,12,300,300);

Bank – library -- package.json{

}

Pom .xml --

Class Ujjwala{

Ujjwala(int x){}

}

Class Bhumika extends Ujjwala{

}

Bhumika b = new Bhumika(); //compilation error

Base class reference variable can store address of child class object

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

14-8-2024

Datastructures – shape of data in memory

Arrays - static fixed -- collection --

Dynamic and elastic -

Traversal,add,remove ,show

Exceptions - program will terminate –

Try{} catch{} /finally{}

Persistence -- storing data in to permanent storage device like disk

Java.io;

Txt,binary files

State of an object /instance --- data stored in the instance variables at a particular time

Serialization - --

Text -- InputStream ->ByteInputStream,DataInputStream,FileInputStream,

OutputStream->FileOutputStream

FileReader,FileWriter –

Interface WindowListener{

Void windowClosing(WindowEvent e);

Void windowClosed(WindowEvent e);

Void windowMaximised(windoeweven t);

}

Class windowHandler extends WindowAdaper{

}

mnuOpen.addActionListener(new ActionListner(){

public void actionPerformed(ActionEvent e){

mnuOpenActionPerformed(e);

}

);