If the child class implements method present in the parent c it is known as method overr

Page No.
Date :

1) Overload a constructor used to initiali a rectangle of length 4 and breadth 5 for using custom parameter.

2) Repeat 1 for sphere.

Chp 10 Inheritance

→ Inheritance is used to borrow properties & methods from an existing class

Phone → Smart Phone

Super class → Sub class

subclass Extends Superclass

Declaring inheritance in java.

Inheritance in java is declared using extends keyword.

Superclass
↓
Subclass ⇒ subclass extends the superclass

Some Examples:

Vehicle	Animal	Animal	Vehicle
↓	↓	↓	↓
Car	Dog	Cat	Truck

When a class inherits from a superclass it inherits parts of superclass methods and fields.

⇒ Java doesn't support multiple inheritance i.e two classes cannot be superclass for sub class.

Code Example

Inherits in java is declared using extends keyword

```
public class Dog extends Animal {  // Inherits Dog from Animal
    //code
}
```

Quick Quiz: Create a class Animal and Derive another class dog from it.

Constructors in Inheritance.

When a Derived class is extends from the Base class, the constructor of the Base class is executed first followed by the constructor of the derived class

⇒ for the following Inheritance hierarchy the constructors are execute in the order. ① → ② → ③

C_1 → Parent
↓
C_2 → child
↓
C_3 → Grand child

constructors execution is top to Bottom order

Constructors during constructor Overloading

When there are multiple constructors in the parent class, the constructor without any parameters is called from the child class.

⇒ If we want to call the constructor with parameters from the parent class, we can use super keyword.

super(a,b); ⇒ calls the constructor from the parent class which takes 2 variables.

this keyword.

this is a way for us to reference an Object of the class which is being created referenced.
↓ available in class.

this.area-2 → this is a reference to current object

Super keyword.

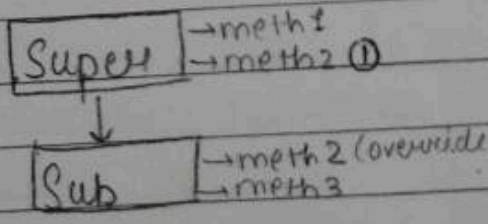
A reference variable used to refer immediate parent class object.

→ can be used to refer immediate parent class instance variable
→ can be used to invoke parent class method
→ can be used to invoke parent class constructors.

Method Overriding
⇒ If the child class implements the same method present in the parent class again, it is known as method overriding.
↳ redefining method of super class in sub class

⇒ When an object of subclass is created and the overrided method is called, the method which has been implemented in the subclass is called & its code is executed.

Dynamic method Dispatch
Consider the following inheritance hierarchy



Scenario 1 → Super obj = new Sub() → Allowed ✓
obj.meth2() → ② is called (Method of obj)
Obj.meth3() → Not allowed ✗

Scenario 2 → Sub obj = new Super() → Not Allowed

To using Abstract class we can create other classes

Page No.
Date :

Chp11 Abstract classes & Interfaces

What does Abstract (class) mean?

⇒ Abstract in english means → existing in thought or as an idea without concrete existence.

Abstract Method

⇒ A method that is declared without an implementation.

```
abstract void moveTo(double x, double y)
```

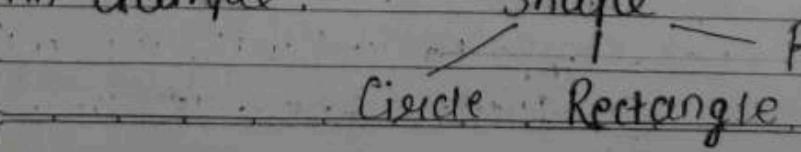
Abstract Class

⇒ If a class includes Abstriact methods, then the class itself must be declared abstract as in:

```
public abstract class PhoneModel {
    abstract void switchOff();
    // more code
}
```

⇒ When an abstract class is subclassed, the subclass usually provides implementation for all of the methods in parent class. If it does not, it must be declared abstract.

An Example.



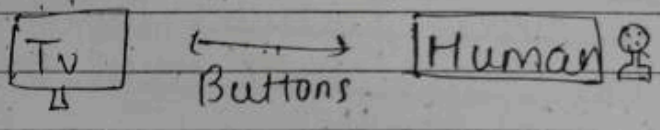
NOTE: It is possible to create reference of an abstract class

⇒ It is not possible to create ~~reference~~ an objects of an abstract class.

⇒ We can also assign reference of an abstract class to the object of a concrete subclass.

~~Int~~ Interfaces in java.

Interface in english is a point where two systems meet & Interact



In java Interface is a group of related methods with Empty bodies

An Example.

```
interface Bicycle {
    void applyBrake(int decrement);
    void speedup(int increment);
}

Class AvonCycle implements Bicycle {
    int speed = 7;
    void applyBrake(int decrement){
        speed = speed - decrement;
    }
```

```
void speed up (int increment) {
        speed = speed + increment;
    }
}
```

Abstract class VS Interfaces

We cant extent multiple abstract classes but we can implement multiple interfaces at a time. Interfaces are meant for dynamic method dispatch & run time polymorphism

Is Multiple Inheritance is allowed in java Multiple Inheritance face problems when there exist methods with same signature in both the super classes.

⇒ Due to such problems, java does not support multiple Inheritance directly but the similar concept can be achieved using interfaces.

⇒ A class can implement multiple Interface and extend a class at the same time.

NOTE: (1) Interfaces in java is a bit like the class but with a significant difference

(2) An Interface can only have method signatures, fields & default methods

3) The class implementing an interface need to define the methods (not fields)

4) You can create a reference of Interface but not the Object.

5) Interface methods are public by default

Default Methods.

⇒ An interface can have static & default methods.

⇒ Default methods enable us to add new functionality to existing interfaces

⇒ This feature is introduced in java 8 to ensure backward compatibility while updating an interface.

⇒ Classes implementing the interface need not implement default methods.

⇒ Interfaces can also include private methods for default methods to use.

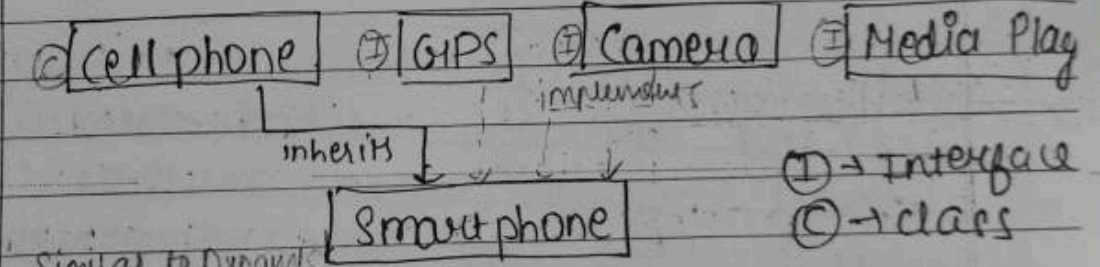
Inheritance in Interfaces.

Interface can extend another Interfaces.

```
public Interface Interface1 {
        void math1();
}
public Interface Interface2 extends Interfac
{
    void meth2();
}
```

Remember that interface cannot implemen another interface, only classes can do that

Polymorphism using Interface



Similar to Dynamic method Dispatch in inheritance

GPS g = new Smartphone(); → can only use GPS method

Smartphone s = new Smartphone(); → can only Smartphone methods

Implementing an Interface forces method implementation.

Page No.
Date :

Ch-11 Practice Set.

1. Create an abstract class pen with methods write() and refill as abstract methods.

2. Use the Pen class from Q1 to create a concrete class fountain pen with additional method changeNib().

3. Create a class Monkey with jump() and bite() methods. Create a class Human which inherits this monkey class and implement Basic Animal interface with eat and sleep methods.

4. Create a class Telephone with ringC), liftC) and disconnect methods an abstract methods create another class SmartTelephone and demostrate polymorphism

5. Demonstrate : polymorphism using math class from Ques 3.

6. Create an Interface TVRemote and ask it to inherit another Interface SmartRemote

7. Create a class TV which implements TVRemote Interface from Q.6.

Page No.
Date :

Chp-12 Packages.

Interpreter Vs Compiler

Interpreter translates one statement at a time into machine code

⇒ Compiler scans the entire program and translates whole of it in Machine code.

Interpreter	Compiler
⇒ one statement at a time.	⇒ Entire program at a time.
⇒ Interpreter is needed Everytime.	⇒ Once compiled is n needed
⇒ Partial execution if error	⇒ No Execution if an error occurs.
⇒ Easy for programmers	⇒ Usually not as ea as Interpreted one

Is java compiled or Interpreted.?

Java is a hybrid language → both compiled & Interpreted.

Java file (harry.java) —compiled (using javac)→ class file (harry.class) → can be used by java Interpreter

↓

bytecode

platform independent

A JVM can be used to Interprete this bytecode

This bytecode can be taken to any platforms (Windows/Mac/Linux) for execution.

Hence, java is platform independent (Write once run everywhere)

Executing a java program

javac Harry.java → compiled
java Harry → Interpreted

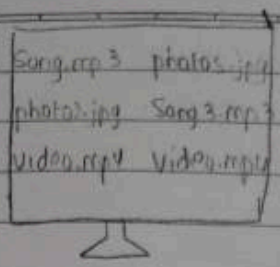
So for the execution of our program was being managed by intellij Idea.

We can download a source code editor like vs code to compile & execute our java programs.

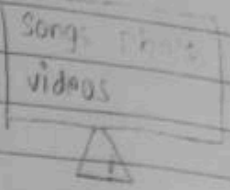
Packages In java.

- A package is used to group related classes.
- Packages help in avoiding name conflicts.
- There are two types of packages:
  1. Built in packages → Java API
  2. User defined packages → Custom packages

this bytecode
any platform
nt C
ogram
deo.
editor
e our
ated
conflicts
:
ci



organized as folders



1. class this.java my.mp3 → organized as package

Song.java Harry.java

Using a java Package :-

import java.lang.* → import everything from java.lang
import java.lang.String → import String from java.lang
S = new Java.lang.String("Harry") → use without importing

Creating a Package :- 65th video

javac Harry.java → creates Harry class
javac -d . Harry.java → creates a package folder
↳ we can keep adding classes to a package like this.

We can also create inner packages by adding "package.inner" as package name.
folder sub folder
These packages once created can be used by other classes.

java -d . abc *.java
all the classes in this folder

Date :

Using Package

import codewithHarry.pack.*;

public class UsingPackage {
    PSVM (String[] args) {
        Sout ("I am using the package");
        Harrybhai g = new Harrybhai();
    }
}

Access Modifiers in java

Access modifiers determine whether other classes can use a particular field or invoke a particular method can be public, private, protected or default (no modifier)

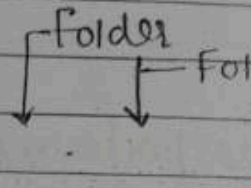
Modifier	class	Package	Subclass	World
Public	Y	Y	Y	Y
Protected	Y	Y	Y	N
Default(no)	Y	Y	N	N
Private	Y	N	N	N

Practice Set

1. Create three classes Calculator, SCalculator and Hybrid Calculator and group them into a package

2. Use a built-in package in java to write a class which displays a message (by using sout) after taking input from the user

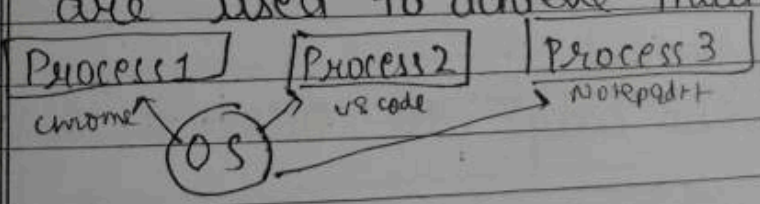
3. create a package in class with three package levels folder; folder1, folder2:

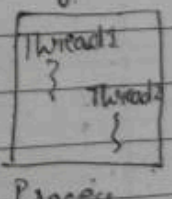


4. Prove that you cannot access default property but can access protected property from the subclass.

Chapter 13 - Multithreading

Multiprocessing and multithreading both are used to achieve multitasking





In a nut shell...

→ Threads used shared memory area
→ Threads ⇒ faster context switching
→ A thread is light weight whereas a process is heavy-weight.

Page No.
Date :

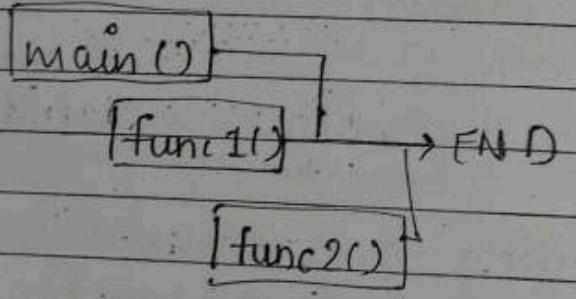
For Example- A word processor can have one thread running in foreground as an editor and another in the background auto saving the document!

Flow of control in java

1. Without Threading:

main() → func1() → func2() → END

2. With Threading: concurrently Execution



Creating a Thread

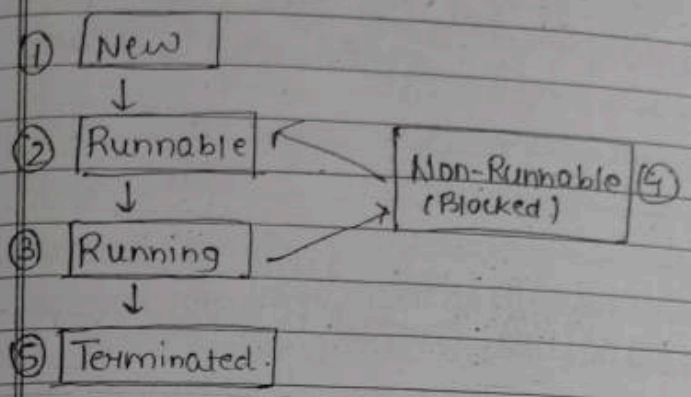
There are two ways to create a thread in java

1. By Extending Thread class
2. By Implementing Runnable Interface

Priority

t5.setPriority(Thread.MAX_PRIORITY);

Life cycle of a thread.



1. New → Instance of thread created which is not yet started by invoking start().
2. Runnable → After invocation of start() & before it is selected to be run by the scheduler.
3. Running → After thread scheduler has selected it.
4. Non-Runnable → Thread alive, not eligible to ru
5. Terminated → run() method has exitted.

⇒ The Thread class

Below are the commonly used constructors of Thread class:

1. Thread()
2. Thread(String name)
3. Thread(Runnable r)
4. Thread(Runnable r, String name)

Methods of Thread Class

Thread class offers a lot of methods such as run(), start(), join(), getPriority(), setPriority etc.

Join() method

allow one thread to wait unit the execution of some other specified thread is complete

Sleep() method

⇒ it useful to put a thread to sleep for a specified amount of time.

⇒ When we put a thread to sleep, the thread scheduler picks and executes another thread in queue.

Interrupt() method:

⇒ A thread in a sleeping or waiting state can be interrupted by another thread with help of interrupt() method.

Practice set

1. Write a program to print "good morning" and "welcome" continuously on the screen in java using Threads.

2. Add a sleep method in welcome thread of question 1 to delay its execution for 2000ms

3. Demonstrate getPriority() and setPriority() methods in java Threads.

4. How do you get state of a given thread in java?

5. How do you get reference to the current thread in java?

Chapter-14

Errors & Exceptions

No matter how smart we are, errors are our constant companions. With practice, we keep getting better at finding & correcting them.

There are three types of errors in java

1. Syntax errors
2. Logical errors
3. Runtime errors → Also called Exceptions.

Syntax Errors

When compiler find something wrong with our program it throws a syntax error

```
int a = 9 → No semicolon, syntax error.
a = a+3;
```

d=4; variable not declared, syntax error.

Logical errors:
A logical error or a bug occurs when a program compiles and runs but does the wrong thing

message delievered wrongly.
wrong time of chats being displayed.
incorrect redirects!

Runtime Errors
Java may sometimes encounter an error while the program is running. These are also called exceptions!

→ These are encountered due to circumstances like bad input and (or) resource constraints.

Ex: user supplies '5' + 8 to a program which adds 2 number;

→ Syntax errors and logical errors are encountered by the programmer where as Runtime errors are encountered by the users.

Exceptions in java
An exception is an event that occurs when a program is executed disrupting the

normal flow of instructions.

→ There are mainly two types of exceptions in java:

1. Checked Exception → Compile time Exceptions (Must be handled)
2. Unchecked Exception → Runtime Exceptions

Commonly occuring Exceptions

→ Following are few commonly occuring exceptions in java.

1. Null pointer Exception
2. Arithmetic Exception
3. ArrayIndexOutofBound Exception
4. IllegalArguement Exception
5. Number format Exception.

Try-Catch Block in java

In java exceptions are managed using try-catch Blocks

Syntax:

```
try {
    //code to try
}
catch (Exception e) {
    // code if Exception
}
```

Page No.
Date:

Handling specific Exceptions

In java, we can handle specific exceptions by typing multiple catch Blocks.

```
try {
    //code
}
catch(IOException e){ → Handles all Exceptions of type IOException
    //code
}
catch(ArithmeticException e){ → Handles all Exceptions of type ArithmeticException
    //code
}
catch(Exception e){ → Handles all other Exceptions
    //code
}
```

Nested try-catch

We can use multiple try-catch blocks as follows

```
try {
    try {
    }
    catch(Exception e) {
    }
}
catch(Exception e) { → Nested try catch Blocks
}
```

Similarly, we can further nest try catch block inside the nested try catch Block

Quick Quiz: Write a java program that allows you to keep accessing an array until a valid index is given by the user.

Exception class in java

We can write our custom Exceptions using Exception class in java

```
public class MyException extends Exception {
        //overridden methods
}
```

The Exception class has following important methods:

1. String toString() → executed when sout(e) is run
2. void printStackTrace() → prints Stack Trace
3. String getMessage() → prints the Exception message

The Throw Keyword.

The throw keyword is used to throw an exception explicitly by the programmer.

```
if(b==0) {
    throw new ArithmeticException("Div by 0");
}
else {
    return a/b;
}
```

In a similar manner, we can throw user defined exceptions:

```
throw new MyException("Exception thrown");
```

The 'throws exception.
The java throws keyword is used to declare an exception. This gives an information to the programmer that there might be an exception so, its better to be prepared with a try catch Block!

```
public void calculate(int a, int b) throws IO
                                        Exception{
    //code
}
```

Java finally Blocks:
finally BLOCK contains the code which is always executed whether the exception is handled or not. It is used to execute code containing instructions to release the system resources, close a connection etc.

Practice set

1. Write a java program to demonstrate syntax logical & runtime errors

Syntax error → int a = 7
logical error → int age = 78;
int year_born = 2000 - 78;
Sout (C/O) → Runtime error.

2) Write a java program that prints "Hello during Arithmetic exception and "Here" during Illegal Exception

3) Write a program that allows you to keep accessing an array until a valid index is given. If max retries exceeds 5 print "Bye"

4) Modify program in Q.3 to throw a custom exception if max retries are reached.

5) Wrap the program in Q.3 inside a method which throws you custom exception.

Advanced Java-1

1. Collections Framework.

A collection represents group of object. Java collections provides classes & interfaces for us to be able to write code quickly & efficiently.

Why do we need collections?

We need collections for efficient storage and better manipulation of data in java. For ex. we use arrays to store integers but what if we want to

- Resize the array?
- Insert an element in between?
- Delete an ~~array~~ element in array?
- Apply certain operations to change this array

How are collections available.

collections in java are available as classes and interfaces following are few commonly used collections in java:

- Arraylist → For variable size collection
- Set → for distinct collection one no one time
- Stack → A LIFO data structure
- HashMap → for storing key-value pairs.

→ Collection class is available in java.util pack

→ Collection class also provides static method for sorting, searching etc.

Array list in java

→ Array list class is the dynamic array found in the java.util.package.

ient storage
ata in java
ove integers

?

ange this array!

le as class
commonly

ion
one time

ers.

a.util package

method for

array

Page No.
Date :

⇒ The size of the normal array cannot be changed, but ArrayList provides us the ability to increase or decrease the si

⇒ ArrayList is slower than the standard array, but it helps to manipulate th data.

Declare: ArrayList<Integer> L1 = new ArrayL<>();

/* creates an ArrayList obje of Integer type */

1. Adding an Element.
- add() method: insert an element in ArrayList
- add(object): Insert an element at the end of ArrayLis
- Add(Index, Object): Inserts Element at the given index.

2. Removing an Element.
- remove(index) to delete or remove an element at a given inde from ArrayLi

3. Checking If an ArrayList contains specific value or no!

contain() method

4. Merging two ArrayLists:

- The Elements of an ArrayList can be merged into another ArrayList with the help of addAll() method.

5. IndexOf

prints the index of 1st occurrence of particular number.

⇒ Returns -1 if the Element is not present in ArrayList.

LinkedList

⇒ The LinkedList class in java provides with the doubly linked list DS.

⇒ Each Element of the LinkedList is known as node.

⇒ Each node points to the address of the next node & its previous node

⇒ Linkedlist are preferred over array list because the insertion & deletion in the linked list can be done in a constant time. But in array if we want to add or delete an element, we need to shift all the other Elements.

st can be
ist with

of particular

not present

rovides with

is known

ess of the

array
deletion
in a constant
t 0 to add
to shift

→ In a linked list, it is impossible to directly access an element because we need to traverse the whole linkedlist to get the desired element.

Array Deque in java.

→ Resizable array + Deque Interface

→ There are no capacity restrictions for ArrayDeque and it provides us the facility to add or remove any element from both sides of queue.

→ faster than LinkedList & Stack.

1. ArrayDeque(): used to create an empty array deque that has the capacity to hold 16 Elements.

2. ArrayDeque(int numElement): used to create empty Array deque that has the capacity to hold specified no. of Elements.

- Methods

1. Insertion at Front: add(), offerFirst()

2. Insertion at last: addLast() / offerLast()

Page No.
Date :

4. Accessing an Element from the head of the deque Array:

getFirst() & peekFirst() methods to get the first element of Deque Array

5. Accessing the last element 'getLast() or peekLast()

7. Removing the First Element : removeFirst() & pollFirst()

- delete an Element from the head of the queue.
- removeFirst() throws an exception if queue is empty.
- pollFirst() returns null if queue is empty

6. Removing the last Element : removeLast() & pollLast()

Hashing In Java

⇒ Hashing is a technique to cover the range of key-value pairs to a range of indices.

⇒ We use hashfunctio
Some value
Let arr = {11, 33, 2
hash = key % 10
Inde

Collision : The has
key value

To avoid collisio
• openAddressing
• Chaining (LinkedLi

Has

⇒

⇒ C
⇒ B
as
me

⇒ Allow

We use hashfunctions to map key to some value.

Let arr = [11, 33, 22, 14].

hash = key % 10
Index

Collision: The hash function may map two key value to a single Index.

To avoid collision
- openAddressing
- chaining (LinkedList)

Hashset

- Hashset class uses a hashtable for storing the Elements.
- It implements Set interface.
- Duplicate values are not allowed.
- Before storing any object, the hashset uses the hashcode() and equals() method to check any duplicate entries.
- Allowe null value.
- Best suited for search operation

1. Hashset(): This constructor is used to create a new empty Hashset that can store 16 Elements and have a load factor of 0.75

Page No.
Date :

Inserting Element
add()

Remove Elements for
Remove()
This method does not throws any exception if the specified Element is not present in Hashset

Checking HashSet is Empty or not.

- isEmpty() method is used to check if there is any object in the Hashset or not.
- This method returns a boolean value

4. Remove all the Elements from Hashset
clear()

5. Printing the size of the Hashset :
size()

Date & Time in java.

Java.time → package for Date & Time in java is from java 8 onwards

Before java 8, java.util package used to hold the date and time classes. Now these classes are deprecated

How java sto
Date is in ja
of a long nu
holds the n
1 jan 1970

Java assume
year which
ssed sinc
used

Quick

Page No.
Date :

How java stores a Date?

- Date in java is stored in the form of a long number. This long number holds the no of miliseconds passed since 1 jan 1970.

- Java assumes that 1900 is the start year which means it calculates years passed since 1900 whenever we asks for years passed.

- System.currentTimeMillis() returns no of miliseconds passed. Once no of ms are calculated, we can calculate minutes, seconds & years passed.

Quick Quiz: Is it safe to store the no of ms in a variable of type long? Yes, pretty safe

```
Sout (Long.MAX_VALUE);
Sout (System.currentTimeMillis());
```

Page No.
Date :

The Date class in java

```
Date d = new Date();
Sout(d);
```

We can also use constructors provided by the Date class.

→ Java Date class has few methods which can be used for ex: getDate(), getDay() etc.

→ All these methods are depreciated

Calendar class in java

Calendar is an abstract class that provides calendar related methods in java.

Calendar.getInstance → returns a calendar instance based on current time.

```
Calendar a = Calendar.getInstance();
a.getTime() → prints Time.
```

Calendar Class Methods.

1) get method is used to get year, date, min, second

2) getTime

3) Other met

Gregorian

This cl
of greg

We can
set Me

Time Z

Time
zones
metho

getA
get
ge

a.get(Calendar.SECOND).
a.get(Calendar.MINUTE)
a.get(Calendar.DATE)
a.get(Calendar.YEAR)

2) getTime method returns a date object

3) Other methods look from java docs.

Gregorian Calendar class

This class is used to create an instance of gregorian calendar.

We can change the year moth & date using set Method:

Time zone

Time zone class is used to create Time zones in java. Some of the important methods of Time zone class are:

getAvailableIDs() → get all the available IDs supp
getDefault() → get the default time zone
getID → get the ID of a time zone