AJ PRACTICAL JOURNAL

Saiganesh

Index:

Practical – 1:

A) Demonstrate Generics class

A.1 Demo Generics

A.2 Implementation of Stack using Generics

B) Demonstrate generics method

B.1 Static generic method

B.2 Non-Static generic method

B.3 Bubble sort using generic method

C) Wildcard

C.1 Unbounded

C.2 Lower bounded

C.3 Upper bounded

C.4 Quicksort

A)Demonstrate Generics class

A.1 Demo Generics

**//Practical 1. WAP to demo a Generic Class**

**package com.generics;**

**class BoxDemo<T>{**

**private T content; //T stands for "Type"**

**public BoxDemo() {**

**content=null;**

**}**

**public BoxDemo(T content) {**

**this.content=content;**

**}**

**public T getContent () {**

**return content;**

**}**

**public void setContent(T content) {**

**this.content = content;**

**}**

**}**

**public class Box\_Generics {**

**public static void main(String[] args) {**

**System.*out*.println("C23101 -- Saiganesh A.V.S.S.N Murthy");**

**//Create a Box to hold an integer**

**BoxDemo<Integer> intBox = new BoxDemo<>();**

**intBox.setContent(42);**

**int intContent = intBox.getContent();**

**System.*out*.println("Integer Content:" +intContent);**

**//Create a Box to hold a string**

**BoxDemo<String> stringBox = new BoxDemo<>();**

**stringBox.setContent("Hello World!");**

**String stringContent = stringBox.getContent();**

**System.*out*.println("String Content:" +stringContent);**

**//Create a Box to hold a Double**

**BoxDemo<Double> doubleBox = new BoxDemo<>();**

**doubleBox.setContent(42.12);**

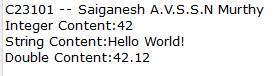
**double doubleContent = doubleBox.getContent();**

**System.*out*.println("Double Content:" +doubleContent);**

**}**

**}**

O/P:



A.2 Implementation of Stack using Generics

**//Define a generic interface for a stack**

**package com.generics;**

**interface Stack<T>{**

**void push (T item);**

**T pop();**

**boolean isEmpty();**

**}**

**//Create a class that implements the generic interface**

**class GenericStack<T> implements Stack<T>{**

**private T[] stackArray;**

**private int top;**

**private int maxSize;**

**@SuppressWarnings ("unchecked")**

**public GenericStack(int maxSize) {**

**this.maxSize = maxSize;**

**this.stackArray= (T[]) new Object[maxSize];**

**this.top = -1;**

**}**

**@Override**

**public void push(T item) {**

**if(top < maxSize -1) {**

**stackArray[++top]=item;**

**}else {**

**System.*out*.println("Stack is full. Cannot push\t"+ item);**

**}**

**}**

**@Override**

**public T pop() {**

**if(!isEmpty()) {**

**return stackArray[top--];**

**}**

**else {**

**System.*out*.println();**

**System.*out*.println("Stack is empty");**

**return null;**

**}**

**}**

**@Override**

**public boolean isEmpty() {**

**return top == -1;**

**}**

**}**

**public class Stack\_Generics {**

**public static void main(String[] args) {**

**System.*out*.println("C23101 -- Saiganesh A.V.S.S.N Murthy");**

**// TODO Auto-generated method stub**

**//create a stack of integers**

**Stack<Integer> intStack = new GenericStack<>(5);**

**intStack.push(10);**

**intStack.push(20);**

**intStack.push(60);**

**intStack.push(23);**

**System.*out*.println();**

**System.*out*.println("Popped item:" + intStack.pop());**

**System.*out*.println();**

**System.*out*.println("Is stack Empty?" + intStack.isEmpty());**

**//create a stack of integers**

**Stack<String> stringStack = new GenericStack<>(5);**

**stringStack.push("DELL");**

**stringStack.push("HP");**

**stringStack.push("LENOVO");**

**stringStack.push("ACER");**

**System.*out*.println();**

**System.*out*.println("Popped item:" + stringStack.pop());**

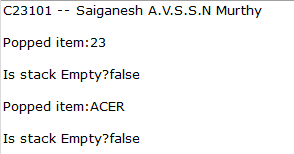
**System.*out*.println();**

**System.*out*.println("Is stack Empty?" + stringStack.isEmpty());**

**}**

**}**

O/P:



B) Demonstrate generics method

B.1 Static generic method

**//To implement a Generic Method.**

**package com.generics;**

**public class Generic\_Method {**

**public static <E> void printArray(E[] inputArray) {**

**for (E element : inputArray) {**

**System.*out*.printf("%s", element);**

**}**

**System.*out*.println();**

**}**

**public static void main(String[] args) {**

**System.*out*.println("C23101 -- Saiganesh A.V.S.S.N Murthy");**

**Integer[] intArray = {1,2,3,5,4};**

**Double[] doubleArray = {1.1,23.4,3.25,2.34,45.15};**

**Character[] charArray = {'H','E','L','L','O'};**

**System.*out*.println("Array integerArray contains:");**

***printArray*(intArray);**

**System.*out*.println("\nArray doubleArray contains:");**

***printArray*(doubleArray);**

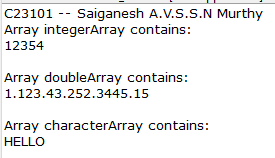
**System.*out*.println("\nArray characterArray contains:");**

***printArray*(charArray);**

**}**

**}**

O/P:



B.2 non-Static generic method

**//To implement Non-Static Generic method.**

**package com.generics;**

**class NS\_GenericM{**

**public <E> void printArray(E[] inputArray) {**

**for (E element : inputArray) {**

**System.*out*.printf("%s", element);**

**}**

**System.*out*.println();**

**}**

**}**

**public class Generic\_NSMethod {**

**public static void main(String[] args) {**

**System.*out*.println("C23101 -- Saiganesh A.V.S.S.N Murthy");**

**NS\_GenericM nsgm = new NS\_GenericM();**

**Integer[] intArray = {1,2,3,5,4};**

**Double[] doubleArray = {1.1,23.4,3.25,2.34,45.15};**

**Character[] charArray = {'H','E','L','L','O'};**

**System.*out*.println("Array integerArray contains:");**

**nsgm.printArray(intArray);**

**System.*out*.println("\nArray doubleArray contains:");**

**nsgm.printArray(doubleArray);**

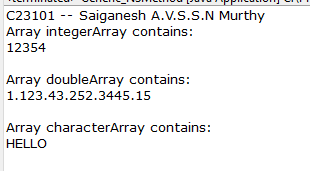
**System.*out*.println("\nArray characterArray contains:");**

**nsgm.printArray(charArray);**

**}**

**}**

O/P:



B.3 Bubble sort using generic method

**package com.generics;**

**public class BubbleSort\_Generics<T extends Comparable<? super T>> {**

**public static void main(String[] args)**

**{**

**System.*out*.println("C23101 -- Saiganesh A.V.S.S.N Murthy");**

**Double[] arrayOfDoubles = {0.35, 0.02, 0.36, 0.82, 0.27, 0.49, 0.41, 0.17, 0.30, 0.89, 0.37, 0.66, 0.82, 0.17, 0.20, 0.96, 0.18, 0.25, 0.37, 0.52};**

**BubbleSort\_Generics<Double> doubleSorter = new BubbleSort\_Generics<>();**

**doubleSorter.bubbleSort(arrayOfDoubles);**

**System.*out*.println(java.util.Arrays.*toString*(arrayOfDoubles));**

**}**

**void bubbleSort(T[] array)**

**{**

**int n = array.length;**

**while (n > 0)**

**{**

**int lastModifiedIndex = 0;**

**for (int currentIndex = 1; currentIndex < n; currentIndex++)**

**{**

**if (array[currentIndex - 1].compareTo(array[currentIndex]) > 0)**

**{**

**T temp = array[currentIndex - 1];**

**array[currentIndex - 1] = array[currentIndex];**

**array[currentIndex] = temp;**

**lastModifiedIndex = currentIndex;**

**}**

**}**

**// save the last modified index so we know not to iterate past it since all proceeding values are sorted**

**n = lastModifiedIndex;**

**}**

**}**

**}**

C) Wildcard

C.1 Unbounded

**package wilcard;**

**import java.util.\*;**

**public class WildcardExample {**

**public static void main(String[] args) {**

**System.*out*.println("C23101 -- Saiganesh A.V.S.S.N Murthy");**

**// TODO Auto-generated method stub**

**List <?> myList = new ArrayList<>();**

**myList.add(null);**

**//myList.add("Hello"); //This results in compile-time error as we have not mentioned a return-type**

**List<String> stringList= new ArrayList<>();**

**stringList.add("Hello");**

**stringList.add("world");**

**List<Integer> intList= new ArrayList<>();**

**intList.add(10);**

**intList.add(5);**

***printListElements*(stringList);**

***printListElements*(intList);**

**}**

**private static void printListElements(List<?> list) {**

**// TODO Auto-generated method stub**

**for(Object item: list) {**

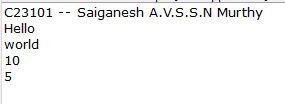
**System.*out*.println(item);**

**}**

**}**

**}**

O/P:



C.2 Lower bounded

package wilcard;

import java.util.ArrayList;

import java.util.List;

public class LowerBoundWC {

private static void addElements(List<? super Integer> list) {

// TODO Auto-generated method stub

for (int i =1;i<=15;i++) {

list.add(i);

}

}

public static void main(String[] args) {

System.out.println("C23101 -- Saiganesh A.V.S.S.N Murthy");

List<Object> objectList = new ArrayList<>();

addElements(objectList);

System.out.println("Object List:" + objectList);

List<Number> numberList= new ArrayList<>();

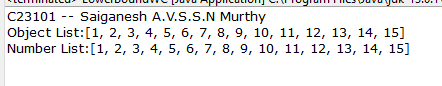
addElements(numberList);

System.out.println("Number List:" + numberList);

}

}

O/P:



C.3 Upper bounded

package wilcard;

import java.util.ArrayList;

import java.util.List;

public class UpperBoundWC {

public static <T extends Comparable <T>> T findMax(List <T> list) {

if (list == null || list.isEmpty()) {

return null;

}

T max = list.get(0);

for (T item : list) {

if(item.compareTo(max)>0) {

max=item;

}

}

return max;

}

public static void main(String[] args) {

// TODO Auto-generated method stub

System.out.println("C23101 -- Saiganesh A.V.S.S.N Murthy");

List<Integer> intList= new ArrayList<>();

intList.add(12);

intList.add(25);

intList.add(55);

List<Double> doubleList= new ArrayList<>();

doubleList.add(12.63);

doubleList.add(22.14);

doubleList.add(89.69);

Integer maxInteger = findMax(intList);

System.out.println("Max Integer:" + maxInteger);

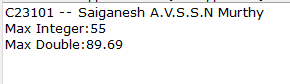
Double maxDouble = findMax(doubleList);

System.out.println("Max Double:" + maxDouble);

}

}

O/P:



C.4 Quicksort