**LAMBDA EXPRESSIONS**

**Practical No 7**

**7.1 Aim: Write java program to using lambda expressions to print “Hello world”**

**Code –**

**package** lambda\_expressions;

**interface** Message{

**public** **void** show();

}

**public** **class** Hello\_world {

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

String msg="Hello World";

//with lambda

Message d2=()->{

System.***out***.println("Message is : "+msg);

};

d2.show();

}

}

**7.2 Aim: Write a java program using lambda expression with single parameter.**

**Code –**

**package** lambda\_expressions;

**interface** Message1{

**public** String say(String name);

}

**public** **class** single\_parameter {

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

{

// Lambda expression with single parameter.

Message1 s1=(name)->{

**return** "Hello, "+name;

}; System.***out***.println(s1.say("Students"));

// You can omit function parentheses

Message1 s2= name ->{

**return** "Hello, "+name;

};

System.***out***.println(s2.say("Students"));

}

}

**7.3 Aim: Write a java program to perform basic mathematical operations (using lambda expressions with multiple parameters)**

**Code –**

**package** lambda\_expressions;

**interface** MathOperation {

**int** operation(**int** a, **int** b); }

**interface** GreetingService {

**void** sayMessage(String message); }

**public** **class** math\_operations {

**private** **int** operate(**int** a, **int** b, MathOperation mathopr)

{

**return** mathopr.operation(a, b);

}

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

{

math\_operations tester = **new** math\_operations();

MathOperation addition=(**int** a, **int** b)->a+b; /\*with type declaration\*/

MathOperation subtraction=(a, b)->a-b; /\*without type declaration\*/

MathOperation multiplication=(**int** a, **int** b)->{ **return** a\*b; };

/\*with return statement and curly braces\*/

MathOperation division=(**int** a, **int** b)->a/b;

/\*without return statement and curly braces\*/

System.***out***.println("10 + 5 = "+tester.operate(10, 5, addition));

System.***out***.println("10 - 5 = "+tester.operate(10, 5, subtraction));

System.***out***.println("10 \* 5 = "+tester.operate(10, 5, multiplication));

System.***out***.println("10 / 5 = "+tester.operate(10, 5, division));

GreetingService greetService1 = message -> System.***out***.println("Hello "+message);

/\*without paranthesis\*/

GreetingService greetService2 = (message) -> System.***out***.println("Hello "+message);

/\*with paranthesis\*/

greetService1.sayMessage("Ramesh");

greetService2.sayMessage("Suresh");

}

}

**7.4 Aim: Write a program to using lambda expressions to calculate the following things –**

1. **Conversion of Fahrenheit to Celsius.**
2. **Conversion to KM to miles.**

**Code 7.4.a –**

**package** lambda\_expressions;

**interface** Tempreture {

**double** calculate(**int** x);

}

**public** **class** temperature\_conversion {

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

Tempreture a = (x) -> {

**double** t = ((x\*9/5)+32);

**return** (t);

};

System.***out***.println(a.calculate(35));

}

}

**Code 7.4.b –**

**package** lambda\_expressions;

**interface** distance {

**double** calculate(**int** x);

}

**public** **class** distance\_conversions {

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

distance a = (x) -> {

**float** miles = (**float**)(0.6213711922 \* x);

**return** (miles);

};

System.***out***.println(a.calculate(100));

}

}

**7.5 Aim: Write a java program to demonstrate lambda expression with or without return keyword.**

**Code –**

**package** lambda\_expressions;

**interface** Addable{

**int** add(**int** a,**int** b);

}

**public** **class** lambda\_with\_without\_return {

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

// Lambda expression without return keyword.

Addable ad1=(a,b)->(a+b);

System.***out***.println(ad1.add(10,20));

// Lambda expression with return keyword.

Addable ad2=(**int** a,**int** b)->{

**return** (a+b);

};

System.***out***.println(ad2.add(100,200));

}

}