Lambda Expression

Question-1

**package** com.lambdaexpression;

**interface** arithmetic

{

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

}

**public** **class** Q1\_LambdaExpression {

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

{

arithmetic addition=(**int** a,**int** b)->(a+b);

System.***out***.println("Addition is :"+addition.operation(7,9));

arithmetic subtraction=(**int** a,**int** b)->(a-b);

System.***out***.println("Subtraction is:"+subtraction.operation(17,12));

arithmetic multiply = (**int** a,**int** b)->(a\*b);

System.***out***.println("Multiply is:"+multiply.operation(20,5));

arithmetic divide =(**int** a,**int** b)->(a/b);

System.***out***.println("Division is:"+divide.operation(100,20));

}

}

Output: Addition is :16

Subtraction is:5

Multiply is:100

Division is:5

Question-2

Order.java

**package** com.lambdaexpression;

**public** **class** Order {

**private** **int** orderId;

**private** **double** price;

**private** Status status;

**public** Order(**int** orderId, **double** price, Status status) {

**this**.orderId = orderId;

**this**.price = price;

**this**.status = status;

}

**public** **int** getOrderId() {

**return** orderId;

}

**public** **void** setOrderId(**int** orderId) {

**this**.orderId = orderId;

}

**public** **double** getPrice() {

**return** price;

}

**public** **void** setPrice(**double** price) {

**this**.price = price;

}

**public** Status getStatus() {

**return** status;

}

**public** **void** setStatus(Status status) {

**this**.status = status;

}

@Override

**public** String toString() {

**return** "Order [orderId=" + orderId + ", price=" + price + ", status=" + status + "]";

}

}

OrderMain.java

**package** com.lambdaexpression;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** OrderMain {

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

List<Order> orderlist = **new** ArrayList<Order>();

orderlist.add(**new** Order(1, 4000, Status.***ACCEPTED***));

orderlist.add(**new** Order(2, 47000, Status.***COMPLETED***));

orderlist.add(**new** Order(3, 25000, Status.***REJECTED***));

orderlist.add(**new** Order(4, 6000, Status.***PENDING***));

orderlist.add(**new** Order(5, 5000, Status.***CANCELLED***));

orderlist.add(**new** Order(7, 12000, Status.***ACCEPTED***));

orderlist.add(**new** Order(6, 18000, Status.***REJECTED***));

orderlist.stream()

.filter(o -> o.getPrice() > 10000)

.filter(o -> o.getStatus()==Status.***ACCEPTED*** || o.getStatus()==Status.***COMPLETED***)

.forEach(o -> System.***out***.println(o));

}}

Status.java

**package** com.lambdaexpression;

**public** **enum** Status {

***COMPLETED***, ***ACCEPTED***, ***REJECTED***, ***PENDING***, ***CANCELLED***;

}

Output: Order [orderId=2, price=47000.0, status=COMPLETED]

Order [orderId=7, price=12000.0, status=ACCEPTED]

Question-3

**package** com.lambdaexpression;

**import** java.util.function.Consumer;

**import** java.util.function.Function;

**import** java.util.function.Predicate;

**import** java.util.function.Supplier;

**public** **class** Q3\_LambdaExpression{

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

Predicate<String> predicate = s -> s.length()==8;

Supplier<String> supplier = () -> "Pankaj";

Consumer<String> consumer= s -> System.***out***.println("String is: " + s);

Function<String,Integer> function = s -> s.length();

System.***out***.println(predicate.test("hello"));

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

consumer.accept("Pankaj");

System.***out***.println(function.apply("Pankaj"));

}

}

Output: false

Pankaj

String is: Pankaj

6

Question-4

**package** com.lambdaexpression;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Q4\_LambdaExpression {

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

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

words.add("Horse");

words.add("Dog");

words.add("Cat");

words.add("Elephant");

words.add("Lion");

words.add("Rabbit");

words.add("Peacock");

words.add("Butterfly");

words.removeIf(w -> w.length()%2 ==1);

System.***out***.println("Even length words are: \n");

System.***out***.println("1st way\n" + words);

System.***out***.println("\n2nd way");

words.stream().forEach(w -> System.***out***.println(w));

}

}

Output: 2nd way

Elephant

Lion

Rabbit

Question-5

**package** com.lambdaexpression;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.function.Consumer;

**public** **class** Q5\_LambdaExpression {

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

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

words.add("Lion");

words.add("Monkey");

words.add("Bat");

words.add("Elephant");

words.add("Tiger");

words.add("Rabbit");

words.add("Peacock");

words.add("Butterfly");

StringBuilder str = **new** StringBuilder("");

Consumer<String> consumer = s -> str.append(s.charAt(0));

words.forEach(w -> consumer.accept(w));

System.***out***.println("Final String is: " + str);

}

}

Output: Final String is: LMBETRPB

Question-6

**package** com.lambdaexpression;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Q6\_LambdaExpression{

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

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

words.add("Horse");

words.add("Dog");

words.add("Cat");

words.add("Elephant");

words.add("Lion");

words.add("Rabbit");

words.add("Peacock");

words.add("Butterfly");

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

words.replaceAll( w -> w.toUpperCase() );

System.***out***.println("\nResult is: " + words);

}

}

Output: [Horse, Dog, Cat, Elephant, Lion, Rabbit, Peacock, Butterfly]

Result is: [HORSE, DOG, CAT, ELEPHANT, LION, RABBIT, PEACOCK, BUTTERFLY]

Question-7

**package** com.lambdaexpression;

**import** java.util.HashMap;

**import** java.util.Map;

**import** java.util.function.BiConsumer;

**public** **class** Q7\_LambdaExpression {

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

Map<Integer, String> words = **new** HashMap<Integer, String>();

words.put(1, "Horse");

words.put(2, "Dog");

words.put(3, "Cat");

words.put(4, "Elephant");

words.put(5, "Tiger");

words.put(6, "Parrot");

words.put(7, "Eagle");

words.put(8, "Monkey");

StringBuilder str = **new** StringBuilder();

BiConsumer<Integer,String> consumer = (key, value) -> str.append(key + value);

words.entrySet().forEach(e -> consumer.accept(e.getKey(), e.getValue()));

System.***out***.println("Final string : " + str);

}

}

Output: Final string : 1Horse2Dog3Cat4Elephant5Tiger6Parrot7Eagle8Monkey

Question-8

**package** com.lambdaexpression;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Q8\_LambdaExpression {

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

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

numberlist.add(10);

numberlist.add(20);

numberlist.add(30);

numberlist.add(40);

numberlist.add(50);

numberlist.add(60);

Thread thread = **new** Thread(() -> numberlist.forEach( num -> System.***out***.println(num)));

System.***out***.println("Number list is: ");

thread.run();

}

}

Output: Number list is:

10

20

30

40

50

60