```
--------------MathOperation.java-----
package com.ameya.intfs;
//@FunctionalInterface
public interface MathOperation {
    int operation(int a,int b);
}
-------------------GreetingService.java----------
package com.ameya.intfs;

public interface GreetingService {
    void sayMessage(String message);
}
-----------------TestLambda.java------------
package com.ameya.test;

import com.ameya.intfs.GreetingService;
import com.ameya.intfs.MathOperation;

public class TestLambda {

    private int operate(int a,int b,MathOperation op) {
        return op.operation(a, b);
    }
    public static void main(String[] args) {
        //Write a class that implements MathOperation
        //Override operation method
        //Create object of the class
        //pass it to operate method

        MathOperation addition = (int a,int b) -> a+b;
        MathOperation subtraction = (a,b) -> a-b;
        MathOperation multiplication=(int a,int b) ->{
            int c=a*b;
            return c;
        };
        MathOperation division=(a,b)->a/b;
        TestLambda tester=new TestLambda();
        System.out.println("SUM :: "+tester.operate(10, 5, addition));
        System.out.println("Diff :: "+tester.operate(10, 5, subtraction));
        System.out.println("Prod :: "+tester.operate(10, 5,
        multiplication));
```

```
        System.out.println("Div :: "+tester.operate(10, 5, division));

        GreetingService gs1=message ->{
            String uMsg=message.toUpperCase();
            System.out.println("Hello "+uMsg);
        };
        gs1.sayMessage("ameya");
    }

}
----------Person.java----------
package com.ameya.test;

public class Person {
    private String firstName;
    private String lastName;
    public Person() {}
    public Person(String firstName,String lastName) {
        this.firstName=firstName;
        this.lastName=lastName;
    }
    @Override
    public String toString() {
        return "Person [firstName=" + firstName + ", lastName=" +
        lastName + "]";
    }

}
----------LambdaDemo2.java----------------
package com.ameya.test;

public class LambdaDemo2 {

    @FunctionalInterface
    public static interface Converter<F,T>{
        T convert(F from);
    }
    static class Something{
        String startsWith(String s) {
            return String.valueOf(s.charAt(0));
        }
```

```
81      }
82      interface PersonFactory<P extends Person>{
83          P create(String firstName,String lastName);
84      }
85      public static void main(String[] args) {
86          Converter<String, Integer>
            intConverter1=(from)->Integer.valueOf(from);
87          Integer converted1=intConverter1.convert("1234");
88          System.out.println(converted1);
89
90          Converter<String,Integer> intConverter2=Integer::valueOf;
91          Integer converted2=intConverter2.convert("123");
92          System.out.println(converted2);
93
94          Something something=new Something();
95          Converter<String,String> strConvert=something::startsWith;
96          String converted3=strConvert.convert("Java in Full Swing");
97          System.out.println(converted3);
98
99          PersonFactory<Person> personFactory=Person::new;
100         Person person=personFactory.create("Ameya", "Joshi");
101         System.out.println(person);
102     }
103
104 }
105 ------------StreamDemo1.java------------
106 package com.ameya.test;
107
108 import java.util.Arrays;
109 import java.util.List;
110 import java.util.stream.Collectors;
111
112 public class StreamDemo1 {
113
114     public static void main(String[] args) {
115         List<Integer> numbers=Arrays.asList(2,3,4,5,2);
116         List<Integer> squares=numbers
117                                     .stream()
118                                     .map(x->x*x)
119                                     .collect(Collectors.toList());
120         System.out.println(squares);
```

```
121         List<String> names=Arrays.asList("Ram","Ameya","Kshiti","Avani");
122         List<String> result=names
123                                     .stream()
124                                     .filter(s->s.startsWith("A"))
125                                     .collect(Collectors.toList());
126         System.out.println(result);
127         List<String> sortedNames=names
128                                                 .parallelStream()
129                                                 .sorted()
130                                                 .collect(Collectors.toList());
131         System.out.println(sortedNames);
132         numbers.stream().map(x->x*x).forEach(y->System.out.println(y));
133         int sum=numbers.parallelStream().filter(x->x%2==0).reduce(0,
            (ans,i)->ans+i);
134                 System.out.println(sum);
135     }
136
137 }
138 ---------------StreamsDemo2.java----------
139 package com.ameya.test;
140
141 import java.util.ArrayList;
142 import java.util.List;
143
144 public class StreamsDemo2 {
145
146     public static void main(String[] args) {
147          List<String> stringCollection = new ArrayList<>();
148             stringCollection.add("ddd2");
149             stringCollection.add("aaa2");
150             stringCollection.add("bbb1");
151             stringCollection.add("aaa1");
152             stringCollection.add("bbb3");
153             stringCollection.add("ccc");
154             stringCollection.add("bbb2");
155             stringCollection.add("ddd1");
156
157
                stringCollection.stream().filter((s)->s.startsWith("a")).forEach(
                System.out::println);
158
```

```
            System.out.println("----------------------------------
            --------");
159
            stringCollection.stream().sorted().filter((s)->s.startsWith("a")).
            forEach(System.out::println);
160
            System.out.println("----------------------------------
            --------");
161
            stringCollection.stream().map(String::toUpperCase).sorted((a,b)
            ->b.compareTo(a)).forEach(System.out::println);
162
            System.out.println("----------------------------------
            --------");
163     }
164
165 }
166 --------------------StringMethods.java----------
167 package com.ameya.test;
168
169 import java.util.regex.Pattern;
170 import java.util.stream.Collectors;
171
172 public class StringMethods {
173
174     public static void main(String[] args) {
175         testJoin();
176         testPatternSplit();
177         testChars();
178     }
179     private static void testJoin() {
180         String str=String.join(":","foobar","foo","bar");
181         System.out.println(str);
182     }
183     public static void testPatternSplit() {
184         String str=Pattern.compile(":")
185                     .splitAsStream("foobar:foo:bar")
186                     .filter(s->s.contains("bar"))
187                     .sorted().collect(Collectors.joining(":"));
188         System.out.println(str);
189     }
```

```
190     private static void testChars() {
191         String str="foobar:foo:bar"
192                         .chars()
193                         .distinct()
194                         .mapToObj(c->String.valueOf((char)c))
195                         .sorted()
196                         .collect(Collectors.joining());
197         System.out.println(str);
198     }
199 }
200 ------------------Student.java----------
201 package com.ameya.test;
202
203 public class Student {
204     private String firstName;
205     private String lastName;
206     private String address;
207     private String city;
208     private double percent;
209
210     public Student() {
211         super();
212         // TODO Auto-generated constructor stub
213     }
214
215     public Student(String firstName, String lastName, String address,
        String city, double percent) {
216         super();
217         this.firstName = firstName;
218         this.lastName = lastName;
219         this.address = address;
220         this.city = city;
221         this.percent = percent;
222     }
223
224     public String getFirstName() {
225         return firstName;
226     }
227     public void setFirstName(String firstName) {
228         this.firstName = firstName;
229     }
```

```
230     public String getLastName() {
231         return lastName;
232     }
233     public void setLastName(String lastName) {
234         this.lastName = lastName;
235     }
236     public String getAddress() {
237         return address;
238     }
239     public void setAddress(String address) {
240         this.address = address;
241     }
242     public String getCity() {
243         return city;
244     }
245     public void setCity(String city) {
246         this.city = city;
247     }
248     public double getPercent() {
249         return percent;
250     }
251     public void setPercent(double percent) {
252         this.percent = percent;
253     }
254
255     @Override
256     public String toString() {
257         return "Student [firstName=" + firstName + ", lastName=" +
            lastName + ", address=" + address + ", city=" + city
258                 + ", percent=" + percent + "]";
259     }
260
261 }
262 ----------------TransformerFunctions.java----------
263 package com.ameya.functions;
264
265 import com.ameya.test.Student;
266
267 public class TransformerFunctions {
268
269     public static Student stringToStudent(String student) {
```

```
        var datas=student.split(" ");
        var studentObj=new Student();
        if(datas!=null) {
            studentObj.setFirstName(datas[0]);
            studentObj.setLastName(datas[1]);
            studentObj.setAddress(datas[2]);
            studentObj.setCity(datas[3]);
            studentObj.setPercent(Double.parseDouble(datas[4]));
            System.out.println(studentObj);
        }
        return studentObj;
    }
}
----------FileApi.java----------
package com.ameya.test;

import java.io.BufferedReader;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.Stream;

import com.ameya.functions.TransformerFunctions;

public class FileApi {

    public static void main(String[] args)throws Exception {
        //testFind();
    //  testReaderLines();
        studentToList();
    }
    private static void studentToList()throws Exception{

        List<Student> studentsList=null;
        studentsList=Files.readString(Paths.get("sampledata.txt"))
                .lines()
                .map(TransformerFunctions::stringToStudent)
                .collect(Collectors.toList());
        System.out.println("-----------------------------------");
```

```
311         System.out.println(studentsList);
312     }
313     private static void testFind() throws Exception{
314         Path start=Paths.get("");
315         try(Stream<Path> stream = Files.find(start, 5,
            (path,attr)->String.valueOf(path).endsWith(".txt")))
316         {
317             String joined =
                stream.sorted().map(String::valueOf).collect(Collectors.joining("
                ;"));
318             System.out.println("find :: "+joined);
319         }
320     }
321     private static void testReaderLines() throws Exception{
322         Path path=Paths.get("sampledata.txt");
323         try(BufferedReader reader=Files.newBufferedReader(path)){
324             long
                cnt=reader.lines().filter(line->line.contains("Pune")).count();
325             System.out.println(cnt);
326         }
327     }
328 }
329
```