```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

import ex11.Person.Gender;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.Set;
import java.util.SortedSet;
import java.util.TreeSet;
import java.util.stream.Collectors;

/**
 *
 * @author cwting
 */
public class CollectorTest {

    public static void main(String[] args) {

        List<Person> persons = Person.createShortList();

        List<String> names = persons.stream()
                .map(Person::getName)
                .collect(ArrayList::new,
                        ArrayList::add,
                        ArrayList::addAll);

        names.forEach(name -> System.out.println(name));
        System.out.println("");

        // using Collectors below
        List<String> nameList = persons.stream()
                .map(Person::getName)
                .collect(Collectors.toList());

        nameList.forEach(name -> System.out.println(name));
        System.out.println("");

        // To obtain a set of unique names (removing duplicates)
        Set<String> nameSet = persons.stream()
                .map(Person::getName)
                .collect(Collectors.toSet());

        nameSet.forEach(name -> System.out.println(name));
        System.out.println("");

        // To obtain a set of unique names sorted in natural order
        SortedSet<String> uniqueSortedNames = persons.stream()
                .map(Person::getName)
                .collect(Collectors.toCollection(TreeSet::new));

        uniqueSortedNames.forEach(name -> System.out.println(name));
        System.out.println("");

```



```java
        // To obtain a map of email to name
        Map<String, String> email2Name = persons.stream()
                .collect(Collectors.toMap(
                        Person::getEmail,       // keyMapper
                        Person::getName));      // valueMapper

        email2Name.forEach((email, name) -> System.out.println(email + " : " + name));
        System.out.println("");

        // To join the name of each person with a comma separator
        String joinedNames = persons.stream().map(Person::getName)
                .collect(Collectors.joining(", "));
        System.out.println(joinedNames);
        System.out.println("");

        // To group people by their gender
        Map<Gender, List<Person>> peopleByGender = persons.stream()
                .collect(Collectors.groupingBy(Person::getGender));
        System.out.println(peopleByGender);
        System.out.println("");

        // To count people by their gender
        Map<Gender, Long> countPeopleByGender = persons.stream()
                .collect(Collectors.groupingBy(Person::getGender, Collectors.counting()));
        System.out.println(countPeopleByGender);
        System.out.println("");

        // To group people by their gender, then by suranme
        Map<Gender, Map<String, List<Person>>> peopleByGenderBySurName = persons.stream()
                .collect(Collectors.groupingBy(Person::getGender,
                        Collectors.groupingBy(Person::getSurName)));
        System.out.println(peopleByGenderBySurName);
        System.out.println("");

        // To partition people by their age
        Map<Boolean, List<Person>> peopleByAge = persons.stream()
                .collect(Collectors.groupingBy(person -> person.getAge() > 50));
        System.out.println(peopleByAge);
        System.out.println("");

    }

}
```

```java
package ex11;

import java.util.Collections;
import java.util.List;
import java.util.Comparator;

/**
 * @author MikeW
 */
public class ComparatorTest {

    public static void main(String[] args) {

        List<Person> personList = Person.createShortList();

        // Sort with Inner Class
        Collections.sort(personList, new Comparator<Person>() {
            public int compare(Person p1, Person p2) {
                return p1.getSurName().compareTo(p2.getSurName());
            }
        });

        System.out.println("=== Sorted Asc SurName ===");
        for (Person p : personList) {
            p.printName();
        }

        // Use Lambda instead
        // Print Asc
        System.out.println("=== Sorted Asc SurName ===");
        Collections.sort(personList, (Person p1, Person p2) ->
p1.getSurName().compareTo(p2.getSurName()));

        for (Person p : personList) {
            p.printName();
        }

        // Note that the type of p1 and p2 are inferred
        // Print Desc
        System.out.println("=== Sorted Desc SurName ===");
        Collections.sort(personList, (p1, p2) ->
p2.getSurName().compareTo(p1.getSurName()));

        for (Person p : personList) {
            p.printName();
        }
    }
}

```



```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

import static ex11.Person.Gender.MALE;
import java.util.List;
import java.util.Optional;

/**
 *
 * @author vanting
 */
public class FindAndMatchStream {

    public static void main(String[] args) {

        List<Person> persons = Person.createShortList();

        // Check if all persons in the list are male
        boolean allMales = persons.stream().allMatch(person -> person.getGender() ==
MALE);

        // Check if any person is at 25
        boolean anyoneAt25 = persons.stream().anyMatch(person -> person.getAge() ==
25);

        // Find the first male
        Optional<Person> firstMale
                = persons.stream()
                        .filter(person -> person.getGender() == MALE)
                        .findFirst();

        System.out.println(allMales);
        System.out.println(anyoneAt25);
        System.out.println(firstMale);

    }

}

```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

import java.util.ArrayList;
import java.util.List;
import java.util.function.Consumer;
import java.util.function.Function;
import java.util.function.Predicate;

/**
 *
 * @author vanting
 */
public class FunctionUtil {

    // Apply an action on each item in a list
    public static <T> void forEach(List<T> list,
            Consumer<? super T> action) {
        for (T item : list) {
            action.accept(item);
        }
    }

    //Apply a filter to a list, returned the filtered list items
    public static <T> List<T> filter(List<T> list,
            Predicate<? super T> predicate) {
        List<T> filteredList = new ArrayList<>();
        for (T item : list) {
            if (predicate.test(item)) {
                filteredList.add(item);
            }
        }

        return filteredList;
    }

    // Map each item of type T in a list to a value of type R
    public static <T, R> List<R> map(List<T> list,
            Function<? super T, R> mapper) {
        List<R> mappedList = new ArrayList<>();
        for (T item : list) {
            mappedList.add(mapper.apply(item));
        }

        return mappedList;
    }
}

```



```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;


import static ex11.Person.Gender.MALE;
import java.util.List;

/**
 *
 * @author vanting
 */
public class FunctionUtilTest {

    public static void main(String[] args) {

        List<Person> list = Person.createShortList();

        // Use forEach() method to print each person in the list
        System.out.println("Original list of persons:");
        FunctionUtil.forEach(list, p -> System.out.println(p));

        // Filter only males
        List<Person> maleList = FunctionUtil.filter(list, p -> p.getGender() ==
MALE);

        System.out.println("\nMales only:");
        FunctionUtil.forEach(maleList, p -> System.out.println(p));

        // Map each person to his/her email
        List<String> emailList = FunctionUtil.map(list,
                p -> p.getEmail());

        System.out.println("\nPersons mapped to their email:");
        FunctionUtil.forEach(emailList, email -> System.out.println(email));

        // Apply an action to each person in the male list
        // Add one to each male's age
        FunctionUtil.forEach(maleList,
                p -> p.setAge(p.getAge() + 1));

        System.out.println(
                "\nMales only after adding 1 to age:");

        FunctionUtil.forEach(maleList, p -> System.out.println(p));
    }
}

```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

/**
 *
 * @author vanting
 */
interface MyInterface {

    /* This is a default method so we need not
     * to implement this method in the implementation
     * classes
     */
    default void newMethod() {
        System.out.println("Newly added default method");
    }

    /* This is a static method. Static method in interface is
     * similar to default method except that we cannot override
     * them in the implementation classes.
     * Similar to default methods, we need to implement these methods
     * in implementation classes so we can safely add them to the
     * existing interfaces.
     */
    static void anotherNewMethod() {
        System.out.println("Newly added static method");
    }

    /* Already existing public and abstract method
     * We must need to implement this method in
     * implementation classes.
     */
    void existingMethod(String str);
}

public class Java8InterfaceExample implements MyInterface {

    // implementing abstract method
    public void existingMethod(String str) {
        System.out.println("String is: " + str);
    }

    public static void main(String[] args) {
        Java8InterfaceExample obj = new Java8InterfaceExample();
        //calling the default method of interface
        obj.newMethod();
        //calling the static method of interface
        MyInterface.anotherNewMethod();
        //calling the abstract method of interface
        obj.existingMethod("Java 8 is easy to learn");

    }
}

```



```java
package ex11;

import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;

/**
 *
 * @author MikeW
 */
public class ListenerTest {

    public static void main(String[] args) {

        JButton testButton = new JButton("Test Button");
        testButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent ae) {
                System.out.println("Click Detected by Anon Class");
            }
        });

        testButton.addActionListener(e -> System.out.println("Click Detected by Lambda Listner"));

        // Swing stuff
        JFrame frame = new JFrame("Listener Test");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.add(testButton, BorderLayout.CENTER);
        frame.pack();
        frame.setVisible(true);

    }
}

```

```java
package ex11;

import java.util.ArrayList;
import java.util.List;

/**
 * @author MikeW
 */
public class Person {

    enum Gender {
        MALE, FEMALE
    }

    private String givenName;
    private String surName;
    private int age;
    private Gender gender;
    private String eMail;
    private String phone;
    private String address;

    public static class Builder {

        private String givenName = "";
        private String surName = "";
        private int age = 0;
        private Gender gender = Gender.FEMALE;
        private String email = "";
        private String phone = "";
        private String address = "";

        public Person.Builder givenName(String givenName) {
            this.givenName = givenName;
            return this;
        }

        public Person.Builder surName(String surName) {
            this.surName = surName;
            return this;
        }

        public Person.Builder age(int val) {
            age = val;
            return this;
        }

        public Person.Builder gender(Gender val) {
            gender = val;
            return this;
        }

        public Person.Builder email(String val) {
            email = val;
            return this;
        }

        public Person.Builder phoneNumber(String val) {
            phone = val;
```



```java
            return this;
        }

        public Person.Builder address(String val) {
            address = val;
            return this;
        }

        public Person build() {
            return new Person(this);
        }
    }

    private Person() {
        super();
    }

    private Person(Person.Builder builder) {
        givenName = builder.givenName;
        surName = builder.surName;
        age = builder.age;
        gender = builder.gender;
        eMail = builder.email;
        phone = builder.phone;
        address = builder.address;

    }

    public String getName() {
        return givenName + " " + surName;
    }

    public String getGivenName() {
        return givenName;
    }

    public String getSurName() {
        return surName;
    }

    public int getAge() {
        return age;
    }

    public Gender getGender() {
        return gender;
    }

    public String getEmail() {
        return eMail;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public void print() {
        System.out.println(
                "\nName: " + givenName + " " + surName + "\n"
                + "Age: " + age + "\n"
```

```
                + "Gender: " + gender + "\n"
                + "eMail: " + eMail + "\n"
                + "Phone: " + phone + "\n"
                + "Address: " + address + "\n"
        );
    }

    public void printName() {
        System.out.println(
                "Name: " + givenName + " " + surName);
    }

    @Override
    public String toString() {
        return "Name: " + givenName + " " + surName + "\n" + "Age: " + age + "
Gender: " + gender + "\n" + "eMail: " + eMail + "\n" + "Address: " + address + "\n";
    }

    public static List<Person> createShortList() {
        List<Person> people = new ArrayList<>();

        people.add(
                new Person.Builder()
                        .givenName("Bob")
                        .surName("Baker")
                        .age(21)
                        .gender(Gender.MALE)
                        .email("bob.baker@example.com")
                        .phoneNumber("201-121-4678")
                        .address("44 4th St, Smallville, KS 12333")
                        .build()
        );

        people.add(
                new Person.Builder()
                        .givenName("Jane")
                        .surName("Jones")
                        .age(25)
                        .gender(Gender.FEMALE)
                        .email("jane.doe@example.com")
                        .phoneNumber("202-123-4678")
                        .address("33 3rd St, Smallville, KS 12333")
                        .build()
        );

        people.add(
                new Person.Builder()
                        .givenName("John")
                        .surName("Baker")
                        .age(25)
                        .gender(Gender.MALE)
                        .email("john.doe@example.com")
                        .phoneNumber("202-123-4678")
                        .address("33 3rd St, Smallville, KS 12333")
                        .build()
        );

        people.add(
                new Person.Builder()
                        .givenName("James")
```



```
                        .surName("Johnson")
                        .age(45)
                        .gender(Gender.MALE)
                        .email("james.johnson@example.com")
                        .phoneNumber("333-456-1233")
                        .address("201 2nd St, New York, NY 12111")
                        .build()
        );

        people.add(
                new Person.Builder()
                        .givenName("Joe")
                        .surName("Baker")
                        .age(67)
                        .gender(Gender.MALE)
                        .email("joebob.bailey@example.com")
                        .phoneNumber("112-111-1111")
                        .address("111 1st St, Town, CA 11111")
                        .build()
        );

        people.add(
                new Person.Builder()
                        .givenName("Phil")
                        .surName("Smith")
                        .age(55)
                        .gender(Gender.MALE)
                        .email("phil.smith@examp;e.com")
                        .phoneNumber("222-33-1234")
                        .address("22 2nd St, New Park, CO 222333")
                        .build()
        );

        people.add(
                new Person.Builder()
                        .givenName("Betty")
                        .surName("Jones")
                        .age(85)
                        .gender(Gender.FEMALE)
                        .email("betty.jones@example.com")
                        .phoneNumber("211-33-1234")
                        .address("22 4th St, New Park, CO 222333")
                        .build()
        );

        return people;
    }

}
```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

/**
 *
 * @author vanting
 */
public class PrimeUtil {

    private long lastPrime = 1L;

    public long next() // instance method
    {
        lastPrime = next(lastPrime); // call static method
        return lastPrime;
    }

    public static long next(long after) {
        long counter = after;
        while (!isPrime(++counter))
         ;
        return counter;
    }

    public static boolean isPrime(long num) {
        if (num <= 1) {
            return false;
        }

        if (num == 2) {
            return true;
        }

        if (num % 2 == 0) {
            return false;
        }

        long maxDivisor = (long) Math.sqrt(num);
        for (int k = 3; k <= maxDivisor; k += 2) {
            if (num % k == 0) {
                return false;
            }
        }

        return true;
    }

}

```



```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

import java.util.stream.Stream;

/**
 *
 * @author vanting
 */
public class PrimeUtilTest {

    public static void main(String[] args) {
        // Print the first 5 prime numbers: 2, 3, 5, 7, 11
        // Data type L for the seed value must be provided.
        // Compiler uses it to infer the data type of stream elements.
        // Use static method next(long)
        Stream.iterate(2L, PrimeUtil::next) // n -> PrimeUtil.next(n)
                .limit(5)
                .forEach(System.out::println);

        // Alternative design
        Stream.iterate(2L, n -> n + 1)
                .filter(PrimeUtil::isPrime) // n -> PrimeUtil.isPrime(n)
                .limit(5)
                .forEach(System.out::println);

        // Design using the Stream.generate(Supplier) method,
        // and skip the first 100 prime numbers
        // use member method primeUtilObj.next()
        Stream.generate(new PrimeUtil()::next)
                .skip(100)
                .limit(5)
                .forEach(System.out::println);

        /*
        Supplier<Long> s = new Supplier()
                           {
                              PrimeUtil pu = new PrimeUtil();
                              public Long get()
                              {
                                 return pu.next();
                              }
                           };

        Stream.generate(s)
              .skip(100)
              .limit(5)
              .forEach(System.out::println);
         */
    }

}

```

```java
package ex11;

/**
 * @author MikeW
 */
public class RunnableTest {

    public static void main(String[] args) {

        System.out.println("=== RunnableTest ===");

        // Anonymous Runnable
        Runnable r1 = new Runnable() {

            @Override
            public void run() {
                System.out.println("Hello world one!");
            }
        };

        // Lambda Runnable
        Runnable r2 = () -> System.out.println("Hello world two!");

        // Run em!
        r1.run();
        r2.run();

    }
}

```



```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

import java.util.DoubleSummaryStatistics;
import java.util.List;

/**
 *
 * @author vanting
 */
public class StatisticsTest {

    public static void main(String[] args) {

        // External iteration
        DoubleSummaryStatistics stats = new DoubleSummaryStatistics();

        List<Person> persons = Person.createShortList();
        for (Person p : persons) {
            stats.accept(p.getAge());
        }

        // Get statistics
        System.out.println("Count: " + stats.getCount());
        System.out.println("Average: " + stats.getAverage());
        System.out.println("Sum: " + stats.getSum());
        System.out.println("Min: " + stats.getMin());
        System.out.println("Max: " + stats.getMax());


        // Alternative design using Stream
        DoubleSummaryStatistics stats2
                = Person.createShortList()
                        .stream()
                        .map(Person::getAge)
                        .collect(DoubleSummaryStatistics::new,
                                DoubleSummaryStatistics::accept,
                                DoubleSummaryStatistics::combine);

        // Get statistics
        System.out.println("Count: " + stats2.getCount());
        System.out.println("Average: " + stats2.getAverage());
        System.out.println("Sum: " + stats2.getSum());
        System.out.println("Min: " + stats2.getMin());
        System.out.println("Max: " + stats2.getMax());

    }
}
```

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ex11;

import java.util.List;
import java.util.Optional;

/**
 *
 * @author vanting
 */
public class TestOptional {

    public static void main(String[] args) {

        List<Person> list = Person.createShortList();
        testAge(list, 10);
        testAge(list, 25);
    }

    public static void testAge(List<Person> list, int age) {
        // If age is not found, output:
        //    Not Found
        // If age is found, output:
        //    name

        Optional<String> result = findAge(list, age);

        if (result.isPresent()) {
            // if Optional has a value
            System.out.println(result.get()); // get the value
        } else {
            System.out.println("Not Found");
        }

        // Alternative implementation using orElse()
        System.out.println(result.orElse("Not Found"));

        // If age is found, output name in upper case
        System.out.println(result.map(String::toUpperCase).orElse("Not Found"));
    }

    static Optional<String> findAge(List<Person> list, int age) {
        for (Person s : list) {
            if (s.getAge() == age) {
                return Optional.of(s.getName());
            }
        }

        return Optional.empty(); // name not found
    }

    static void testFn(List<Person> list) {

        Optional<String> result = findAge(list, 1234);

```



```java
    }

}

```