МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

***Циклова комісія програмування***

**Робочий план**

з курсу «Java - програмування»

Практична робота №7

Солом’яного Ярослава Сергійовича

*ПІБ студента*

студента групи **2П-18**

|  |  |  |
| --- | --- | --- |
| **Вид роботи** | **Дата** | **Оцінка** |
| *Пром. контроль* |  |  |
| *Залік* |  |  |

Викладач Марченко С. В.

Черкаси-2021

Завдання 1. Виконайте вправи, представлені за посиланням.

<http://binkurt.blogspot.com/2017/10/exercises-to-study-java-stream-api.html>

Лістинг 1. Class Movie

package com.example.domain;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Movie {  
 private int id;  
 private String title;  
 private int year;  
 private String imdb;  
 private List<Genre> genres;  
 private List<Director> directors;  
  
 //region constructor, setters/getters, hashCode, equals, and toString  
 {  
 genres = new ArrayList<>();  
 directors = new ArrayList<>();  
 }  
  
 public Movie() {  
 }  
  
 public Movie(int id, String title, int year, String imdb) {  
 this.id = id;  
 this.title = title;  
 this.year = year;  
 this.imdb = imdb;  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
  
 public int getYear() {  
 return year;  
 }  
  
 public void setYear(int year) {  
 this.year = year;  
 }  
  
 public String getImdb() {  
 return imdb;  
 }  
  
 public void setImdb(String imdb) {  
 this.imdb = imdb;  
 }  
  
 public List<Genre> getGenres() {  
 return genres;  
 }  
  
 public void setGenres(List<Genre> genres) {  
 this.genres = genres;  
 }  
  
 public List<Director> getDirectors() {  
 return directors;  
 }  
  
 public void setDirectors(List<Director> directors) {  
 this.directors = directors;  
 }  
  
 @Override  
 public int hashCode() {  
 final int prime = 31;  
 int result = 1;  
 result = prime \* result + id;  
 return result;  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 if (this == obj)  
 return true;  
 if (obj == null)  
 return false;  
 if (getClass() != obj.getClass())  
 return false;  
 Movie other = (Movie) obj;  
 if (id != other.id)  
 return false;  
 return true;  
 }  
  
 @Override  
 public String toString() {  
 return "Movie [title=" + title + ", year=" + year  
 + "]";  
 }  
 //endregion  
}

Лістинг №2.Class Genre

package com.example.domain;  
  
/\*\*  
 \*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Genre {  
 private int id;  
 private String name;  
  
 public Genre() {  
 }  
  
 public Genre(int id, String name) {  
 this.id = id;  
 this.name = name;  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 @Override  
 public int hashCode() {  
 final int prime = 31;  
 int result = 1;  
 result = prime \* result + id;  
 return result;  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 if (this == obj)  
 return true;  
 if (obj == null)  
 return false;  
 if (getClass() != obj.getClass())  
 return false;  
 Genre other = (Genre) obj;  
 if (id != other.id)  
 return false;  
 return true;  
 }  
  
 @Override  
 public String toString() {  
 return "Genre [id=" + id + ", name=" + name + "]";  
 }  
  
}

Лістинг №3. Class Director

package com.example.domain;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Director {  
 private int id;  
 private String name;  
 private String imdb;  
 private List<Movie> movies= new ArrayList<>();  
  
  
 public Director() {  
 }  
  
 public Director(int id, String name, String imdb) {  
 this.id = id;  
 this.name = name;  
 this.imdb = imdb;  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getImdb() {  
 return imdb;  
 }  
  
 public void setImdb(String imdb) {  
 this.imdb = imdb;  
 }  
  
 public List<Movie> getMovies() {  
 return movies;  
 }  
  
 public void setMovies(List<Movie> movies) {  
 this.movies = movies;  
 }  
  
 @Override  
 public int hashCode() {  
 final int prime = 31;  
 int result = 1;  
 result = prime \* result + id;  
 return result;  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 if (this == obj)  
 return true;  
 if (obj == null)  
 return false;  
 if (getClass() != obj.getClass())  
 return false;  
 Director other = (Director) obj;  
 if (id != other.id)  
 return false;  
 return true;  
 }  
  
 @Override  
 public String toString() {  
 return "Director [id=" + id + ", name=" + name + ", imdb=" + imdb + "]";  
 }  
}

Лістинг 4. Class Country

package com.example.domain;  
  
import java.util.ArrayList;  
import java.util.List;  
  
/\*\*  
 \*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Country {  
 private String code;  
 private String name;  
 private String continent;  
 private double surfaceArea;  
 private int population;  
 private double gnp;  
 private int capital;  
 private List<City> cities;  
 {  
 cities = new ArrayList<>();  
 }  
  
 public Country() {  
 }  
  
 public Country(String code, String name, String continent, int population,  
 double surfaceArea, double gnp, int capital) {  
 this.code = code;  
 this.name = name;  
 this.continent = continent;  
 this.surfaceArea = surfaceArea;  
 this.population = population;  
 this.capital = capital;  
 this.gnp = gnp;  
 }  
  
 public String getCode() {  
 return code;  
 }  
  
 public void setCode(String code) {  
 this.code = code;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getContinent() {  
 return continent;  
 }  
  
 public void setContinent(String continent) {  
 this.continent = continent;  
 }  
  
 public double getSurfaceArea() {  
 return surfaceArea;  
 }  
  
 public void setSurfaceArea(double surfaceArea) {  
 this.surfaceArea = surfaceArea;  
 }  
  
 public double getGnp() {  
 return gnp;  
 }  
  
 public void setGnp(double gnp) {  
 this.gnp = gnp;  
 }  
  
 public int getCapital() {  
 return capital;  
 }  
  
 public void setCapital(int capital) {  
 this.capital = capital;  
 }  
  
 public void setPopulation(int population) {  
 this.population = population;  
 }  
  
 public int getPopulation() {  
 return population;  
 }  
  
 public List<City> getCities() {  
 return cities;  
 }  
  
 @Override  
 public int hashCode() {  
 final int prime = 31;  
 int result = 1;  
 result = prime \* result + ((code == null) ? 0 : code.hashCode());  
 return result;  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 if (this == obj)  
 return true;  
 if (obj == null)  
 return false;  
 if (getClass() != obj.getClass())  
 return false;  
 Country other = (Country) obj;  
 if (code == null) {  
 if (other.code != null)  
 return false;  
 } else if (!code.equals(other.code))  
 return false;  
 return true;  
 }  
  
 @Override  
 public String toString() {  
 return "Country [ name=" + name + ", population="  
 + population + "]";  
 }  
  
}

Лістинг №5.Class City

package com.example.domain;  
  
/\*\*  
 \*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class City {  
 private int id;  
 private String name;  
 private int population;  
 private String countryCode;  
  
 public City() {  
 }  
  
 public City(int id, String name, String countryCode, int population) {  
 this.id = id;  
 this.name = name;  
 this.population = population;  
 this.countryCode = countryCode;  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getPopulation() {  
 return population;  
 }  
  
 public void setPopulation(int population) {  
 this.population = population;  
 }  
  
 public String getCountryCode() {  
 return countryCode;  
 }  
  
 public void setCountryCode(String countryCode) {  
 this.countryCode = countryCode;  
 }  
  
 @Override  
 public int hashCode() {  
 final int prime = 31;  
 int result = 1;  
 result = prime \* result + id;  
 return result;  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 if (this == obj)  
 return true;  
 if (obj == null)  
 return false;  
 if (getClass() != obj.getClass())  
 return false;  
 City other = (City) obj;  
 if (id != other.id)  
 return false;  
 return true;  
 }  
  
 @Override  
 public String toString() {  
 return "City [id=" + id + ", name=" + name + ", population="  
 + population + ", countryCode=" + countryCode + "]";  
 };  
  
}

Лістинг №6. Вправа 1.

package com.example.exercise;  
import java.util.Comparator;  
import java.util.List;  
import java.util.Optional;  
import java.util.stream.Collectors;  
  
import com.example.dao.CountryDao;  
import com.example.dao.InMemoryWorldDao;  
import com.example.domain.City;  
  
/\*\*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Exercise1 {  
 public static void main(String[] args) {  
 CountryDao countryDao = InMemoryWorldDao.getInstance();  
 List<City> highPopulatedCitiesOfCountries = countryDao.findAllCountries()  
 .stream()  
 .map(country -> country.getCities().stream().max(Comparator.comparing(City::getPopulation)))  
 .filter(Optional::isPresent)  
 .map(Optional::get)  
 .collect(Collectors.toList());  
 highPopulatedCitiesOfCountries.forEach(System.out::println);  
 }  
  
}

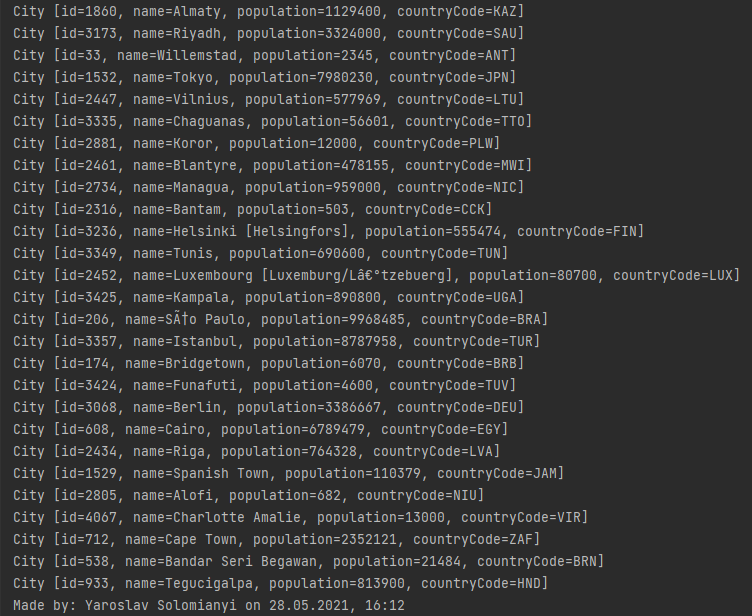


Рис 1. Результат роботи програми №1.

Лістинг №7. Вправа 2.

package com.example.exercise;  
  
import java.text.SimpleDateFormat;  
import java.util.Comparator;  
import java.util.Date;  
import java.util.Map;  
import java.util.Map.Entry;  
import java.util.Optional;  
import java.util.function.BiConsumer;  
import java.util.function.Consumer;  
import java.util.function.Predicate;  
import java.util.stream.Collector;  
import java.util.stream.Collectors;  
  
import com.example.dao.CountryDao;  
import com.example.dao.InMemoryWorldDao;  
import com.example.domain.City;  
  
/\*\*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Exercise2 {  
  
 public static void main(String[] args) {  
 // Find the most populated city of each continent  
 CountryDao countryDao = InMemoryWorldDao.getInstance();  
 final Predicate<Entry<String, Optional<City>>> isPresent = entry -> entry.getValue().isPresent();  
 final BiConsumer<String, Optional<City>> printEntry =  
 (k,v) -> {  
 City city = v.get();  
 System.out.println(k + ": City [ name= " + city.getName() + ", population= " + city.getPopulation() + " ]");  
 };  
 Collector<City, ?, Map<String, Optional<City>>> groupingHighPopulatedCitiesByContinent = Collectors.groupingBy(city -> countryDao.findCountryByCode(city.getCountryCode()).getContinent(), Collectors.maxBy(Comparator.comparing(City::getPopulation)));  
 Map<String, Optional<City>> highPopulatedCitiesByContinent = countryDao.findAllCountries()  
 .stream()  
 .flatMap(country -> country.getCities().stream())  
 .collect(groupingHighPopulatedCitiesByContinent);  
 highPopulatedCitiesByContinent.forEach(printEntry);  
 printTimeStamp();  
 }  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
}

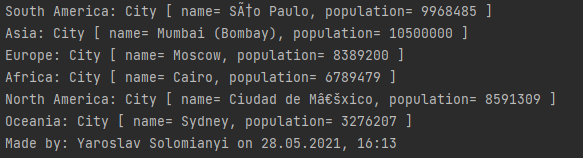


Рис 2. Результат роботи для завдання 2.

Лістинг 8. Завдання 3.

package com.example.exercise;  
  
import java.text.SimpleDateFormat;  
import java.util.Collection;  
import java.util.Date;  
import java.util.List;  
import java.util.Map;  
import java.util.stream.Collectors;  
  
import com.example.domain.Director;  
import com.example.domain.Movie;  
import com.example.service.InMemoryMovieService;  
import com.example.service.MovieService;  
  
/\*\*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Exercise3 {  
  
 public static void main(String[] args) {  
 // Find the number of movies of each director  
 MovieService movieService = InMemoryMovieService.getInstance();  
 Collection<Movie> movies = movieService.findAllMovies();  
 Map<String, Long> directorMovieCounts =  
 movies.stream()  
 .map(Movie::getDirectors)  
 .flatMap(List::stream)  
 .collect(Collectors.groupingBy(Director::getName, Collectors.counting()));  
 directorMovieCounts.entrySet().forEach(System.out::println);  
 printTimeStamp();  
 }  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
}

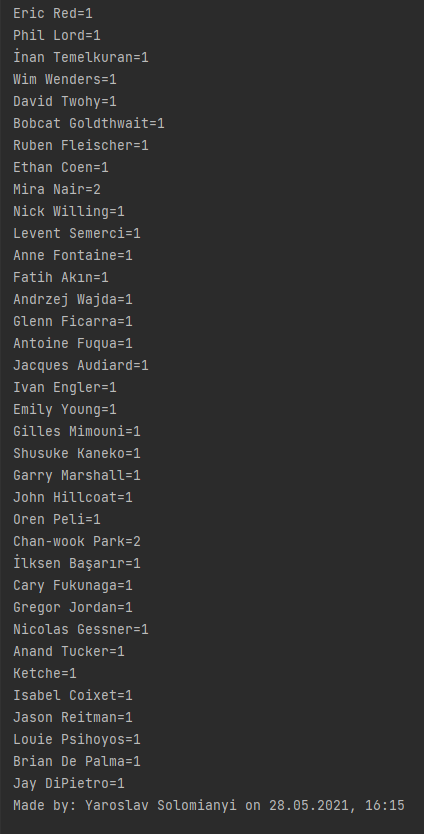


Рис 3. Результат виконання завдання 3.

Лістинг 9. Завдання 4.

package com.example.exercise;  
  
import java.text.SimpleDateFormat;  
import java.util.Collection;  
import java.util.Date;  
import java.util.Map;  
import java.util.stream.Stream;  
  
import com.example.domain.Director;  
import com.example.domain.Genre;  
import com.example.domain.Movie;  
import com.example.service.InMemoryMovieService;  
import com.example.service.MovieService;  
  
import static java.util.stream.Collectors.\*;  
  
/\*\*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Exercise4 {  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
 public static void main(String[] args) {  
 // Find the number of genres of each director's movies  
 MovieService movieService = InMemoryMovieService.getInstance();  
 Collection<Director> directors = movieService.findAllDirectors();  
 Stream<DirectorGenre> stream =  
 directors.stream()  
 .flatMap(director -> director.getMovies()  
 .stream()  
 .map(Movie::getGenres)  
 .flatMap(Collection::stream)  
 .map(genre -> new DirectorGenre(director, genre))  
 .collect(toList()).stream()  
 );  
 Map<Director, Map<Genre, Long>> directorGenreList =  
 stream.collect(  
 groupingBy(  
 DirectorGenre::getKey,  
 groupingBy(DirectorGenre::getValue, counting())  
 )  
 );  
 directorGenreList.forEach(  
 (k1,v1) -> {  
 System.out.println(k1.getName());  
 v1.forEach( (k2,v2) -> {  
 System.out.println(String.format("\t%-12s: %2d", k2.getName(), v2));  
 });  
 System.out.println();  
 printTimeStamp();  
 }  
 );  
 }  
  
}  
  
class DirectorGenre implements Map.Entry<Director, Genre> {  
 private Director director;  
 private Genre genre;  
  
 public DirectorGenre(Director director, Genre genre) {  
 this.director = director;  
 this.genre = genre;  
 }  
  
 @Override  
 public Director getKey() {  
 return director;  
 }  
  
 @Override  
 public Genre getValue() {  
 return genre;  
 }  
  
 @Override  
 public Genre setValue(Genre genre) {  
 this.genre = genre;  
 return genre;  
 }  
  
}

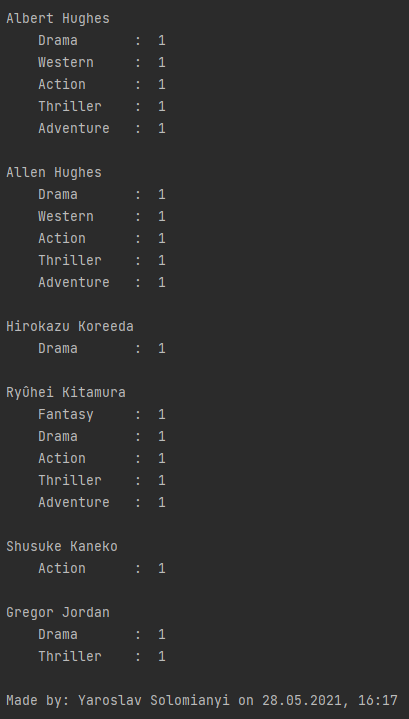


Рис 4. Результат виконання завдання 4

Лістинг 10. Завдання 5

package com.example.exercise;  
  
import com.example.dao.CityDao;  
import com.example.dao.CountryDao;  
import com.example.dao.InMemoryWorldDao;  
import com.example.domain.City;  
import com.example.domain.Country;  
  
import java.text.SimpleDateFormat;  
import java.util.Date;  
import java.util.Objects;  
import java.util.Optional;  
  
import static java.lang.System.out;  
import static java.util.Comparator.comparing;  
import static java.util.stream.Collectors.maxBy;  
  
/\*\*  
 \* @author Binnur Kurt (binnur.kurt@gmail.com)  
 \*/  
public class Exercise5 {  
  
 public static void main(String[] args) {  
 // Find the most populated capital  
 CountryDao countryDao = InMemoryWorldDao.getInstance();  
 CityDao cityDao = InMemoryWorldDao.getInstance();  
 Optional<City> capital = countryDao.findAllCountries()  
 .stream()  
 .map(Country::getCapital)  
 .map(cityDao::findCityById)  
 .filter(Objects::nonNull)  
 .collect(maxBy(comparing(City::getPopulation)));  
 capital.ifPresent(out::println);  
 printTimeStamp();  
 }  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
}



Рис 5. Результат виконання завдання 5.

Завдання 2. Запустіть та опишіть приклади роботи паралельних стрімів.

<https://mkyong.com/java8/java-8-parallel-streams-examples/>

Лістинг 11. Завдання 1.

package Ex\_2;  
  
import java.util.stream.IntStream;  
  
public class Ex\_1 {  
 public static void main(String[] args) {  
  
 System.out.println("Normal...");  
  
 IntStream range = IntStream.rangeClosed(1, 10);  
 range.forEach(System.out::println);  
  
 System.out.println("Parallel...");  
  
 IntStream range2 = IntStream.rangeClosed(1, 10);  
 range2.parallel().forEach(System.out::println);  
 }  
}

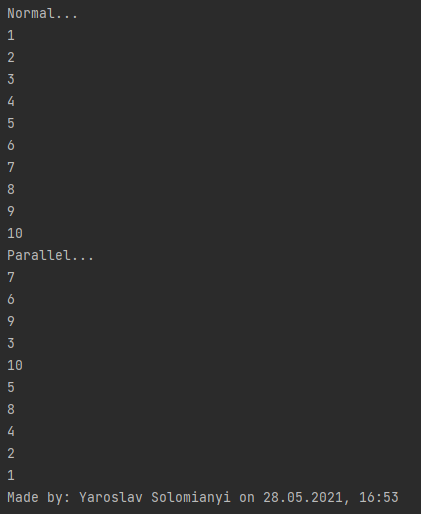


Рис 6. Результат роботи завдання 1.

В цьому завданні використовується бібліотека Stream, з якої ми беремо стрім для потоку данних типу Int, також за допомогою методу .parallel() запускаємо виконання в паралельному режимі.

Лістинг 12. Завдання 2.

package Ex\_2;  
  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
  
public class ex\_2 {  
 public static void main(String[] args) {  
  
 System.out.println("Normal...");  
  
 List<String> alpha = getData();  
 alpha.stream().forEach(System.out::println);  
  
 System.out.println("Parallel...");  
  
 List<String> alpha2 = getData();  
 alpha2.parallelStream().forEach(System.out::println);  
 printTimeStamp();  
  
 }  
 private static List<String> getData() {  
  
 List<String> alpha = new ArrayList<>();  
  
 int n = 97; // 97 = a , 122 = z  
 while (n <= 122) {  
 char c = (char) n;  
 alpha.add(String.valueOf(c));  
 n++;  
 }  
 return alpha;  
 }  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
  
}

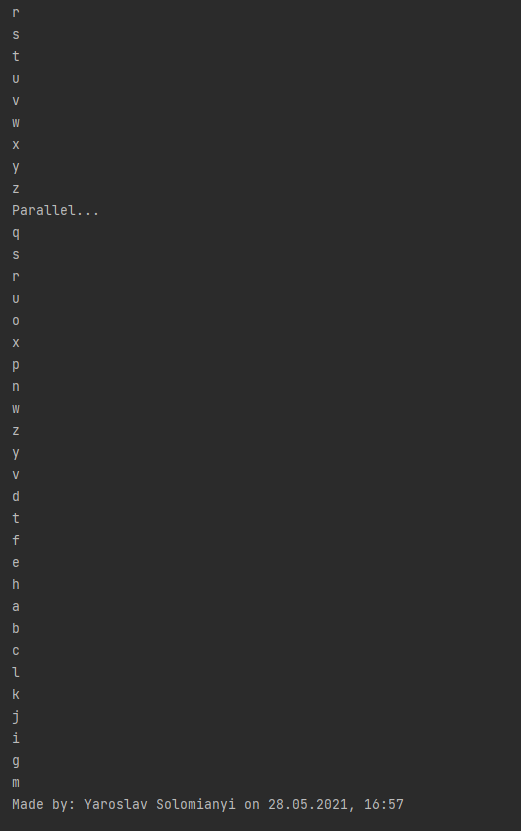


Рис 7. Результат виконання завдання 2.

В цьому завданні представлений метод типу List, який повертає нам букву, також можна спостерігати, що коли просто виводимо алфавіт, він виводиться по порядку, якщо використовувати паралельно потоки, то він виводиться не по порядку.

Лістинг 13. Завдання 3.

package Ex\_2;  
  
import java.text.SimpleDateFormat;  
import java.util.Date;  
import java.util.stream.IntStream;  
  
public class ex\_3 {  
 public static void main(String[] args) {  
  
 System.out.println("Normal...");  
  
 IntStream range = IntStream.rangeClosed(1, 10);  
 System.out.println(range.isParallel()); // false  
 range.forEach(System.out::println);  
  
 System.out.println("Parallel...");  
  
 IntStream range2 = IntStream.rangeClosed(1, 10);  
 IntStream range2Parallel = range2.parallel();  
 System.out.println(range2Parallel.isParallel()); // true  
 range2Parallel.forEach(System.out::println);  
 }  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
}

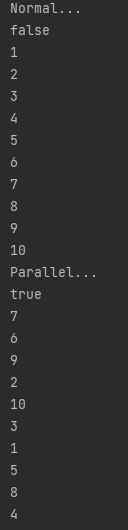


Рис 8. Результат роботи програми 3

В цьому завданні можна спостерігати, як відбувається перевірка чи іде паралельне виконання коду.

Наприклад в першому випадку нам поверне false, томущо ми ніде не викликали .parallel(), а коли 2 раз викликали, то все получилось і ми отримали true;

Лістинг 14. Завдання 4.

package Ex\_2;  
  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
import java.util.stream.IntStream;  
import java.util.stream.Stream;  
  
public class ex\_4 {  
 public static void main(String[] args) {  
  
 long count = Stream.iterate(0, n -> n + 1)  
 .limit(1\_000\_000).parallel()  
 .filter(ex\_4::isPrime)  
 .peek(x -> System.out.format("%s\t", x))  
 .count();  
 System.out.println("\nTotal: " + count);  
 printTimeStamp();  
 }  
 public static boolean isPrime(int number) {  
 if (number <= 1) return false;  
 return !IntStream.rangeClosed(2, number / 2).anyMatch(i -> number % i == 0);  
 }  
 public static void printTimeStamp() {  
 final String sname = "Yaroslav Solomianyi";  
 Date currentDate = new Date();  
 SimpleDateFormat dateFormat = null;  
 dateFormat = new SimpleDateFormat();  
 System.out.println("Made by: " + sname + " on " + dateFormat.format(currentDate));  
 }  
  
}

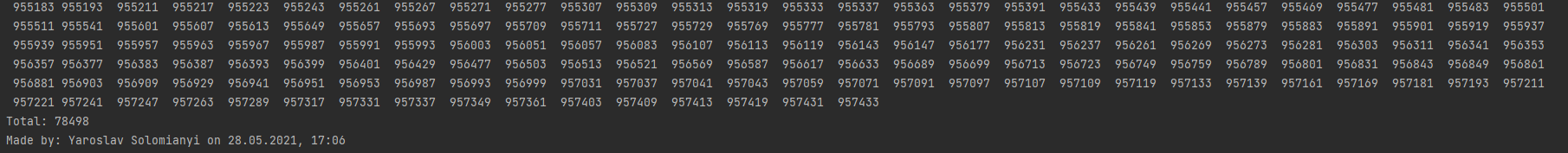


Рис 9. Результат виконання завдання 4

В цьому завданні, ми паралельно виконуємо вивід від 0 до мільйону, використовуючи лямбда вирази, вказуємо як хочемо виводити паралельно чи ні за допомогою .limit().parallel або .limit()

Лістинг 15. Завдання 5.

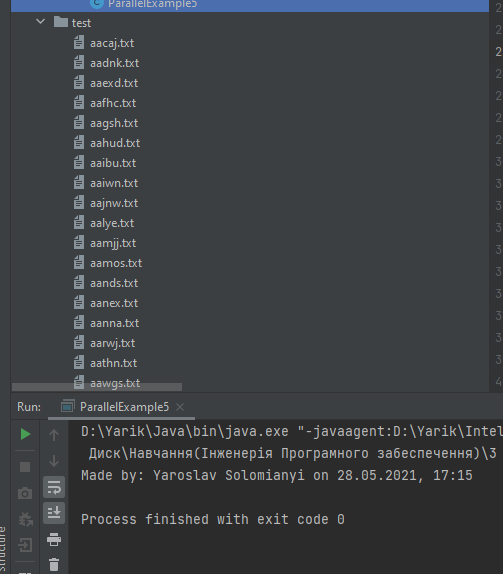


Рис 10. Результат завдання 5

В цьому завданні генеруємо 10000 співробітників з класу Employee та записуємо їх паралельно у файли, перед цим створючи список в який будемо генерувати цих робітників.