/\*

generates an array of 1000 random integers, prompts the user to enter an index between 0 and 1000, and prints the value at that index if it's within bounds; otherwise, it prints "Out of Bounds."

@Author Jack Usery

@Assignment 4 Qn 2

@Date 11/07/2023

\*/

import java.util.InputMismatchException;

import java.util.Random;

import java.util.Scanner;

public class assign4Qn2JackUsery {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Random random = new Random();

int[] size = new int[1000];

for (int i = 0; i < size.length; i++) {

size[i] = random.nextInt(1000);

}

try {

System.out.print("Enter an index between 0 and 1000: ");

int index = scanner.nextInt();

if (index < 0 || index > 1000) {

throw new ArrayIndexOutOfBoundsException();

} else {

System.out.println("The number at index " + index + " is " + size[index]);

}

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("Out of Bounds");

}

finally {

scanner.close();

}

}

}







/\*

This program creates a Person class that has a name, address, phone number, and email address.

@Author: Jack Usery

@Assignment: 4 Qn 3

@Date: 11/09/2020

\*/

public class Person {

private String name;

private String address;

private String phoneNumber;

private String emailAddress;

// constructor

public Person() {

name = "";

address = "";

phoneNumber = "";

emailAddress = "";

}

public Person(String name, String address, String phoneNumber, String emailAddress) {

this.name = name;

this.address = address;

this.phoneNumber = phoneNumber;

this.emailAddress = emailAddress;

}

// accessors

public String getName() {

return name;

}

public String getAddress() {

return address;

}

public String getPhoneNumber() {

return phoneNumber;

}

public String getEmailAddress() {

return emailAddress;

}

// mutators

public void setName(String name) {

this.name = name;

}

public void setAddress(String address) {

this.address = address;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber = phoneNumber;

}

public void setEmailAddress(String emailAddress) {

this.emailAddress = emailAddress;

}

@Override

public String toString() {

return "Person [name = " + name + "]";

}

}

/\*

This is the PartTimeEmployee class. It extends the Person class and adds a classStatus and payRate

@Author: Jack Usery

@Assignment: 4 Qn 3

@Date: 11/09/2020

\*/

public class PartTimeEmployee extends Person{

public static final String classStatusRegular = "Regular";

public static final String classStatusSeasonal = "Seasonal";

public static final String classStatusOnCall = "On-Call";

public static final String classStatusContractor = "Contractor";

private String classStatus;

private double payRate;

// Constructors

public PartTimeEmployee() {

classStatus = "";

payRate = 0.0;

}

public PartTimeEmployee(String name, String address, String phoneNumber, String emailAddress, String classStatus, double payRate) {

super(name, address, phoneNumber, emailAddress);

this.classStatus = classStatus;

this.payRate = payRate;

}

// Accessors

public String getClassStatus() {

return classStatus;

}

public double getPayRate() {

return payRate;

}

// Mutators

public void setClassStatus(String classStatus) {

this.classStatus = classStatus;

}

public void setPayRate(double payRate) {

this.payRate = payRate;

}

// toString

@Override

public String toString() {

return "PartTimeEmployee [name = " + getName() + "]";

}

}

/\*

This program creates a FullTimeEmployee class that extends the Person class and adds an office, salary, and date hired.

@Author: Jack Usery

@Assignment: 4 Qn 3

@Date: 11/09/2020

\*/

public class FullTimeEmployee extends Person {

private String office;

private double salary;

private String dateHired;

// Constructors

public FullTimeEmployee() {

office = "";

salary = 0.0;

dateHired = "";

}

public FullTimeEmployee(String name, String address, String phoneNumber, String emailAddress, String office, double salary, String dateHired) {

super(name, address, phoneNumber, emailAddress);

this.office = office;

this.salary = salary;

this.dateHired = dateHired;

}

// Accessors

public String getOffice() {

return office;

}

public double getSalary() {

return salary;

}

public String getDateHired() {

return dateHired;

}

// Mutators

public void setOffice(String office) {

this.office = office;

}

public void setSalary(double salary) {

this.salary = salary;

}

public void setDateHired(String dateHired) {

this.dateHired = dateHired;

}

// toString

@Override

public String toString() {

return "FullTimeEmployee [name = " + getName() + "]";

}

}

/\*

This is the junior class that extends the full time employee class. It has a mentor and a rank.

@Author: Jack Usery

@Assignment: 4 Qn 3

@Date: 11/09/2020

\*/

public class Junior extends FullTimeEmployee {

private String mentor;

private String rank;

// constructor

public Junior() {

mentor = "";

rank = "";

}

public Junior(String name, String address, String phoneNumber, String emailAddress, String office, double salary, String dateHired, String mentor, String Rank) {

super(name, address, phoneNumber, emailAddress, office, salary, dateHired);

this.mentor = mentor;

this.rank = Rank;

}

// accessors

public String getMentor() {

return mentor;

}

public String getRank() {

return rank;

}

// mutators

public void setRank(String Rank) {

this.rank = rank;

}

public void setMentor(String mentor) {

this.mentor = mentor;

}

// toString

@Override

public String toString() {

return "Junior [name = " + getName() + "]";

}

}

/\*

This program creates a Senior class that extends the FullTimeEmployee class and adds a list of juniors and company stocks.

@Author: Jack Usery

@Assignment: 4 Qn 3

@Date: 11/09/2020

\*/

import java.util.ArrayList;

public class Senior extends FullTimeEmployee {

private ArrayList<Junior> juniorsTeam;

private double companyStocks;

// Constructors

public Senior() {

juniorsTeam = new ArrayList<Junior>();

companyStocks = 0.0;

}

public Senior(String name, String address, String phoneNumber, String emailAddress, String office, double salary, String dateHired, ArrayList<Junior> juniorsTeam, double companyStocks) {

super(name, address, phoneNumber, emailAddress, office, salary, dateHired);

this.juniorsTeam = juniorsTeam;

this.companyStocks = companyStocks;

}

// Accessors

public ArrayList<Junior> getJuniorsTeam() {

return juniorsTeam;

}

public double getCompanyStocks() {

return companyStocks;

}

// Mutators

public void setJuniorsTeam(ArrayList<Junior> juniorsTeam) {

this.juniorsTeam = juniorsTeam;

}

public void setCompanyStocks(double companyStocks) {

this.companyStocks = companyStocks;

}

// toString

@Override

public String toString() {

return "Senior [name = " + getName() + "]";

}

}

/\*

This is the Test class that tests the other classes.

@Author: Jack Usery

@Assignment: 4 Qn 3

@Date: 11/09/2023

\*/

import java.util.ArrayList;

public class Test {

public static void main(String[] args) {

Person person1 = new Person();

person1.setName("Jack");

person1.setAddress("1234 Main St.");

person1.setPhoneNumber("123-456-7890");

person1.setEmailAddress("jack@gmail.com");

System.out.println(person1.toString());

PartTimeEmployee parttimeEmployee1 = new PartTimeEmployee();

parttimeEmployee1.setName("John");

parttimeEmployee1.setAddress("1111 Main St.");

parttimeEmployee1.setPhoneNumber("111-111-1111");

parttimeEmployee1.setEmailAddress("john@gmail.com");

parttimeEmployee1.setClassStatus(PartTimeEmployee.classStatusSeasonal);

parttimeEmployee1.setPayRate(10.00);

System.out.println(parttimeEmployee1.toString());

FullTimeEmployee fulltimeEmployee1 = new FullTimeEmployee();

fulltimeEmployee1.setName("Jane");

fulltimeEmployee1.setAddress("2222 Main St.");

fulltimeEmployee1.setPhoneNumber("222-222-2222");

fulltimeEmployee1.setEmailAddress("jane@gmail.com");

fulltimeEmployee1.setOffice("Office 1");

fulltimeEmployee1.setSalary(50000.00);

System.out.println(fulltimeEmployee1.toString());

ArrayList<Junior> juniorTeam = new ArrayList<>();

juniorTeam.add(new Junior("Junior 1", "3333 Main St.", "333-333-3333", "junior1@gmail.com", "Office 2", 40000.00, "01/01/2020", "Mentor 1", "Rank 1"));

System.out.println(juniorTeam.toString());

Senior senior1 = new Senior();

senior1.setName("Senior 1");

senior1.setAddress("4444 Main St.");

senior1.setPhoneNumber("444-444-4444");

senior1.setEmailAddress("Senior1@gmail.com");

senior1.setOffice("Office 3");

senior1.setSalary(60000.00);

senior1.setDateHired("01/01/2020");

senior1.setJuniorsTeam(juniorTeam);

senior1.setCompanyStocks(10000.00);

System.out.println(senior1.toString());

}

}

A black background with white text

Description automatically generated

/\*

This code creates a window with a button that when clicked will display a message

@Author: Jack Usery

@Assignment: 4 Qn 4

@Date: 11\9\2023

\*/

package com.example.demo;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.stage.Stage;

import java.io.IOException;

public class assign4Qn4JackUsery extends Application {

@Override

public void start(Stage stage) throws IOException {

FXMLLoader fxmlLoader = new FXMLLoader(assign4Qn4JackUsery.class.getResource("hello-view.fxml"));

Scene scene = new Scene(fxmlLoader.load(), 320, 240);

stage.setTitle("Hello Jack! My First JavaFX");

stage.setScene(scene);

stage.show();

}

public static void main(String[] args) {

launch();

}

}

A screenshot of a computer

Description automatically generated