Федеральное государственное автономное образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| институт |
| Кафедра Информатики |
| кафедра |

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

|  |
| --- |
| Создание простого консольного приложения на языке Java |
| Тема |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Преподаватель | |  |  | А. Н. Пупков |
|  |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ19-17/1б, 031939590 |  |  | В. С. Марков |
|  | номер группы, зачетной книжки | подпись, дата |  | инициалы, фамилия |

Красноярск 2021

1 Цель

Ознакомиться с механизмом наследования в языке Java. Повторить основные языковые конструкции языка Java. Результат выполнения выводится на терминал/консоль. Программа должна быть устойчива к некорректному пользовательскому вводу.

2 Задачи

Задание данной практической работы состоит из следующих частей: - разработка программы в соответствии с вариантом; - оформление отчета в соответствии с СТО.

3 Описание задачи

В ходе работы использовался вариант № 15. Для каждого варианта имеется набор из четырех сущностей. Необходимо выстроить иерархию наследования. В каждом классе (базовом и производных) должно быть минимум одно числовое и одно текстовое поле. При вводе числовых параметров обязательна проверка на число и на диапазон (даже если число может быть любое, проверку необходимо реализовать). Для всех классов должны быть реализованы конструкторы (по умолчанию, с параметрами), методы equals(), hashCode(), toString(); Реализовать консольное Java-приложение, которое имеет простейшее пользовательское меню, состоящее как минимум из следующих пунктов:

1. добавить новый элемент. (Элементы должны добавляться в коллекцию элементов типа базового класса. Необходимо предусмотреть возможность добавления любого объекта производного класса в данную коллекцию.);
2. удалить элемент по индексу;
3. вывод всех элементов в консоль;
4. вравнение двух элементов на равенство (по индексам).
5. Завершение работы приложения.

4 Листинги кода

Ниже будут приведены листинги кода.

Листинг 1 – файл classes/Animal.java

package Сlasses;  
  
import java.util.Objects;  
  
public class Animal {  
 protected static final boolean *SUCSESS* = true;  
 protected static final boolean *FAIL* = false;  
  
 protected String code;  
 protected int age;  
  
 private static final int *max\_age* = 25;  
  
 public Animal(){  
 this.code = "";  
 this.age = 0;  
 }  
  
 public void set\_code(String code){  
 this.code = code;  
 }  
  
 public String get\_code(){  
 return this.code;  
 }  
  
 public boolean set\_age(int age){  
 if (age >= 0 && age <= *max\_age*){  
 this.age = age;  
 return *SUCSESS*;  
 }  
 else{  
 return *FAIL*;  
 }  
 }  
  
 public String to\_string(){  
 return String.*format*("UNKNOWN ANIMAL\n->Code: %s\n->Age: %d\n", code, age);  
 }  
  
 public boolean equals(Animal a){  
 return (this == a);  
 }  
  
 public int hashCode(){  
 return Objects.*hash*(code, age);  
 }  
}

Конец листинга 1

Листинг 2 – файл classes/Mammal.java

package Сlasses;  
  
public class Mammal extends Animal{  
  
 private static final int *max\_age* = 150;  
  
 @Override  
 public boolean set\_age(int age){  
 if (age >= 0 && age <= *max\_age*){  
 this.age = age;  
 return *SUCSESS*;  
 }  
 else{  
 return *FAIL*;  
 }  
 }  
  
 @Override  
 public String to\_string() {  
 return String.*format*("UNKNOWN MAMMAL\n->Code: %s\n->Age: %d\n", code, age);  
 }  
}

Конец листинга 2

Листинг 3 – файл classes/Bird.java

package Сlasses;  
  
public class Bird extends Animal{  
  
 private static final int *max\_age* = 10;  
  
  
 public Bird(){  
 this.code = "";  
 this.age = 0;  
 }  
 @Override  
 public boolean set\_age(int age){  
 if (age >= 0 && age <= *max\_age*){  
 this.age = age;  
 return *SUCSESS*;  
 }  
 else{  
 return *FAIL*;  
 }  
 }  
  
 @Override  
 public String to\_string() {  
 return String.*format*("UNKNOWN BIRD\n->Code: %s\n->Age: %d\n", code, age);  
 }  
}

Конец листинга 3

Листинг 4 – файл classes/SeaMammal.java

package Сlasses;  
  
import javax.security.auth.login.FailedLoginException;  
  
public class SeaMammal extends Mammal{  
  
 private static final int *max\_age* = 25;  
  
 public SeaMammal(){  
 this.code = "";  
 this.age = 0;  
 }  
  
 @Override  
 public boolean set\_age(int age){  
 if (age >= 0 && age <= *max\_age*){  
 this.age = age;  
 return *SUCSESS*;  
 }  
 else{  
 return *FAIL*;  
 }  
 }  
  
 @Override  
 public String to\_string() {  
 return String.*format*("UNKNOWN SEA MAMMAL\n->Code: %s\n->Age: %d\n", code, age);  
 }  
}

Конец листинга 4

Листинг 5 – файл classes/Controller.java

package Сlasses;  
  
import java.util.ArrayList;  
import java.util.Scanner;  
  
public class Controller {  
  
  
  
 public static void run(){  
  
 int choise = 1;  
 ArrayList<Animal> animals = new ArrayList<Animal>();  
 Scanner in = new Scanner(System.*in*);  
  
 while (choise != 0){  
 System.*out*.println("1. Create animal");  
 System.*out*.println("2. Create mammal");  
 System.*out*.println("3. Create bird");  
 System.*out*.println("4. Create sea mammal");  
 System.*out*.println("5. Print the animal's list");

Продолжение листинга 5

System.*out*.println("6. Search by code");  
 System.*out*.println("7. Compare animals");  
 System.*out*.println("0. Exit");  
 try {  
 choise = in.nextInt();  
 } catch (NumberFormatException nfe) {  
 System.*out*.println("NaN");  
 break;  
 }  
  
 String temp\_str = "";  
 int temp\_int;  
  
 switch(choise){  
  
 case 1:  
 Animal new\_animal = new Animal();  
 System.*out*.println("Input code:");  
 temp\_str = in.next().toString().toUpperCase();  
 System.*out*.println("Input age (int):");  
 try {  
 temp\_int = in.nextInt();  
 }  
 catch (Exception e) {  
 System.*out*.println("NaN");  
 break;  
 }  
 new\_animal.set\_code(temp\_str);  
 if(new\_animal.set\_age(temp\_int)){  
 animals.add(new\_animal);  
 }  
 else{  
 System.*out*.println("Age error");  
 }  
 break;  
  
 case 2:  
 Animal new\_mammal = new Mammal();  
 System.*out*.println("Input code:");  
 temp\_str = in.next().toString().toUpperCase();  
 System.*out*.println("Input age (int):");  
 try {  
 temp\_int = in.nextInt();  
 }  
 catch (Exception e) {  
 System.*out*.println("NaN");  
 break;  
 }  
 new\_mammal.set\_code(temp\_str);  
 if(new\_mammal.set\_age(temp\_int)){  
 animals.add(new\_mammal);  
 }  
 else{  
 System.*out*.println("Age error");  
 }  
 break;  
 case 3:  
 Animal new\_bird = new Bird();  
 System.*out*.println("Input code:");  
 temp\_str = in.next().toString().toUpperCase();  
 System.*out*.println("Input age (int):");  
 try {  
 temp\_int = in.nextInt();  
 }

Продолжение листинга 5

catch (Exception e) {  
 System.*out*.println("NaN");  
 break;  
 }  
 new\_bird.set\_code(temp\_str);  
 if(new\_bird.set\_age(temp\_int)){  
 animals.add(new\_bird);  
 }  
 else{  
 System.*out*.println("Age error");  
 }  
 break;  
  
 case 4:  
 Animal new\_sea\_mammal = new Bird();  
 System.*out*.println("Input code:");  
 temp\_str = in.next().toString().toUpperCase();  
 System.*out*.println("Input age (int):");  
 try {  
 temp\_int = in.nextInt();  
 }  
 catch (Exception e) {  
 System.*out*.println("NaN");  
 break;  
 }  
 new\_sea\_mammal.set\_code(temp\_str);  
 if(new\_sea\_mammal.set\_age(temp\_int)){  
 animals.add(new\_sea\_mammal);  
 }  
 else{  
 System.*out*.println("Age error");  
 }  
 break;  
 case 5:  
 for (int i = 0; i < animals.size(); i++){  
 System.*out*.println(animals.get(i).to\_string());  
 }  
 break;  
 case 6:  
 System.*out*.println("Input code:");  
 temp\_str = in.next();  
  
 for(int i = 0; i < animals.size(); i++){  
 if (animals.get(i).get\_code().equals(temp\_str.toUpperCase())){  
 System.*out*.println(animals.get(i).to\_string());  
 }  
 if (i == (animals.size() - 1) && !(animals.get(i).get\_code().equals(temp\_str.toUpperCase()))){  
 System.*out*.println("No Results\n");  
 }  
 }  
 break;  
 case 7:  
 Animal a1 = null;  
 Animal a2 = null;  
 System.*out*.println("Input the first id:");  
 temp\_int = in.nextInt();  
 a1 = animals.get(temp\_int);  
 System.*out*.println("Input the second id:");  
 temp\_int = in.nextInt();  
 a2 = animals.get(temp\_int);  
  
 if (a1.hashCode() == a2.hashCode()){

Продолжение листинга 5

System.*out*.println("MATCH\n");  
 }  
 else{  
 System.*out*.println("NOT A MATCH\n");  
 }  
 break;  
 case 0:  
 break;  
 }  
 }  
 }  
}

Конец листинга 5

Листинг 6 – файл Main.java

import Сlasses.\*;  
  
import java.util.ArrayList;  
import java.util.InputMismatchException;  
import java.util.Scanner;  
  
public class Main {  
 public static void main (String[] args){  
 Controller.*run*();  
 }  
}

Конец листинга 6

5 Вывод

В результате выполнения практической работы были получены навыки по работе с наследованием в Java и повторены основные языковые конструкции.