Day6\_Answers:

package day\_6\_Assignment;

/\*Object Casting with Inheritance

1.Define an Animal class with a method makeSound().

2.Define subclass Dog:

oOverride makeSound() (e.g. "Woof!").

oAdd method fetch().

3.In main:

Dog d = new Dog();

Animal a = d; // upcasting

a.makeSound();\*/

class Animal{

void makeSound() {

System.***out***.println("Animal makes sound...");

}

}

class Dog extends Animal{

void makeSound() {

System.***out***.println("Dog barks");

}

void fetch() {

System.***out***.println("Fetch method of dog class");

}

}

public class Main {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

Dog d=new Dog();

Animal a=d;//upcasting

a.makeSound();

}

}

Output:

Dog barks

2. package day\_6\_Assignment;

/\*Mini‑Project – Temperature Converter

1.Prompt user for a temperature in Celsius (double).

2.Convert it to Fahrenheit:

double fahrenheit = celsius \* 9/5 + 32;

3.Then cast that fahrenheit to int for display.

4.Print both the precise (double) and truncated (int) values, and comment on precision loss.\*/

import java.util.Scanner;

public class Mini\_Project {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

System.***out***.println("Temperature Converter!!!");

Scanner sc=new Scanner(System.***in***);

System.***out***.println("Enter temp in celsius : ");

double cel=sc.nextDouble();

double fahrenheit = cel \* 9/5 + 32;

int temp=(int)fahrenheit;

System.***out***.println("Temp in double : "+fahrenheit);

System.***out***.println("Temp in int : "+temp);

}

}

Output:

Temperature Converter!!!

Enter temp in celsius :

78

Temp in double : 172.4

Temp in int : 172

3. package day\_6\_Assignment;

/\*1.Write a program to:

oRead an int value from user input.

oAssign it to a double (implicit widening) and print both.

oRead a double, explicitly cast it to int, then to short, and print results—demonstrate truncation or overflow.\*/

import java.util.Scanner;

public class Program1 {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=new Scanner(System.***in***);

System.***out***.println("Enter n: ");

int n=sc.nextInt();

double d=n;//widening

System.***out***.println("n : "+n);

System.***out***.println("d : "+d);

System.***out***.println("Enterd 2: ");

double d2=sc.nextDouble();

int n2=(int)d2;//narrowing

System.***out***.println("d2 : "+d2);

System.***out***.println("n2 : "+n2);

}

}

Output:

Enter n:

6

n : 6

d : 6.0

Enterd 2:

8

d2 : 8.0

n2 : 8

4. package day\_6\_Assignment;

/\*Convert an int to String using String.valueOf(...),

\* then back with Integer.parseInt(...). Handle NumberFormatException.

\*/

public class Program2 {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

int n=5;

System.***out***.println("Integer n : "+n);

String s=String.*valueOf*(n);

System.***out***.println("n as String : "+s);

try {

int n2=Integer.*parseInt*("abc");

System.***out***.println("n2 as int parsed from String s : "+n2);

}catch(NumberFormatException e) {

System.***err***.println("Pass int values only as it cause NumberFormatException if you pass string");

}

System.***out***.println("Program ends");

}

}

Output:

Integer n : 5

n as String : 5

Pass int values only as it cause NumberFormatException if you pass string

Program ends

5. package day\_6\_Assignment;

/\*Compound Assignment Behaviour

1.Initialize int x = 5;.

2.Write two operations:

x = x + 4.5; // Does this compile? Why or why not?

x += 4.5; // What happens here?

3.Print results and explain behavior in comments (implicit narrowing, compile error vs. successful assignment).\*/

public class Program3 {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

int x=5;

//x=x+4.5;it show error because 4.5 of double type .

System.***out***.println("Implicit narrowing of x=4+4.5 as x+=4.5");

x+=4.5;//compiles successfully as implicit += operator includes an implicit cast.Java automatically converts the result to int before assigning it to x.

System.***out***.println("x : "+x);

}

}

Output:

Implicit narrowing of x=4+4.5 as x+=4.5

x : 9

6. package day\_6\_Assignment;

/\*Days of the Week

Define an enum DaysOfWeek with seven constants. Then in main(), prompt the user to input a day name and:

•Print its position via ordinal().

Confirm if it's a weekend day using a switch or if-statement.\*/

import java.util.Scanner;

public class Program4 {

enum *DaysOfWeek*{***MONDAY***,***TUESDAY***,***WEDNESDAY***,***THURSDAY***,***FRIDAY***,***SATURDAY***,***SUNDAY***}

public static void main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=new Scanner(System.***in***);

System.***out***.println("Enter any DaysOfWeek : ");

String s=sc.nextLine();

*DaysOfWeek* day=*DaysOfWeek*.*valueOf*(s);

if(day==*DaysOfWeek*.***SATURDAY***||day==*DaysOfWeek*.***SUNDAY***) {

System.***out***.println("Weekend");

}

else {

System.***out***.println("Weekday");

}

System.***out***.println("Position via ordinal : "+day.ordinal());//Like indexing the enum type

}

}

Output:

Enter any DaysOfWeek :

MONDAY

Weekday

Position via ordinal : 0

7. package day\_6\_Assignment;

import java.util.Scanner;

/\* Compass Directions

Create an enum Direction with the values NORTH, SOUTH, EAST, WEST. Write code to:

•Read a Direction from a string using valueOf().

Use switch or if to print movement (e.g. “Move north”).

Test invalid inputs with proper error handling.\*/

public class Program5 {

enum *Direction*{***NORTH***, ***SOUTH***, ***EAST***, ***WEST***}

public static void main(String[] args) {

// **TODO** Auto-generated method stub

Scanner sc=new Scanner(System.***in***);

System.***out***.println("Enter any Direction : ");

try {

String direction=sc.nextLine();

*Direction* d=*Direction*.*valueOf*(direction);

switch(d) {

case ***NORTH***:

System.***out***.println("Move NORTH");

break;

case ***SOUTH***:

System.***out***.println("Move SOUTH");

break;

case ***EAST***:

System.***out***.println("Move EAST");

break;

case ***WEST***:

System.***out***.println("Move WEST");

break;

}

}catch(Exception e) {//IllegalArgumentExcetion

System.***out***.println(e);

}

}

}

Output:

Enter any Direction :

NORTH

Move NORTH

8. package day\_6\_Assignment;

public class Throw\_exp {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

String s=null;

if(s==null) {

throw new NullPointerException("Name must not be null");

}

System.***out***.println("program end");

}

}

Output:

Exception in thread "main" java.lang.NullPointerException: Name must not be null at day\_6\_Assignment.Throw\_exp.main(Throw\_exp.java:9)

9. package day\_6\_Assignment;

public class Throws\_exp {

void add(int a,int b) {

System.***out***.println("Addition : "+(a+b));

}

void div(int a,int b) throws ArithmeticException {

System.***out***.println("Div : "+(a/b));

}

public static void main(String[] args) {

// **TODO** Auto-generated method stub

Throws\_exp t=new Throws\_exp();

t.add(10,20);

t.div(50,10);

try {

t.div(10, 0);

}catch(ArithmeticException e) {

System.***out***.println("divide by zero");

}

}

}

Output:

Addition : 30

Div : 5

divide by zero

10. package day\_6\_Assignment;

import java.io.\*;

public class MultiExceptionDemo {

public static void main(String[] args) {

try {

BufferedReader reader = new BufferedReader(new FileReader("data.txt"));

int number = Integer.*parseInt*(reader.readLine());

int result = 100 / number;

System.***out***.println("Result: " + result);

reader.close();

} catch (FileNotFoundException e) {

System.***out***.println("File not found.");

} catch (IOException e) {

System.***out***.println("Problem reading the file.");

} catch (NumberFormatException e) {

System.***out***.println("Invalid number format.");

} catch (ArithmeticException e) {

System.***out***.println("Division by zero.");

} finally {

System.***out***.println("Execution completed.");

}

}

}

Output:

File not found.

Execution completed.