COMP 2522 Midterm Practice Pack

Some things you could possibly choose to study before the exam

202330

Good luck, and have fun!

```java
package ca.bcit.comp2522.midterm.prep;

/**
 * Alpha.
 */
class Alpha {
    /**
     * Foos.
     */
    public void foo() {
        System.out.println("Alpha");
    }
}

/**
 * Beta.
 */
public class Beta extends Alpha {
    /**
     * Foos.
     */
    protected void foo(){
        System.out.println("Beta");
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        Alpha a;
        a = new Beta();
        a.foo();
    }
}
```

1. **This will not compile. Why not? Fix the error without changing the main method.**

```java
/**
 * Gamma.
 */
public class Gamma {

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        Alpha a;
        a = new Beta();
        a.foo();
    }
}
```

2. If this class is in the same package as the previous classes, will it compile? If so what is the output? If not, why not?

```java
package ca.bcit.comp2522.midterm.prep;

/**
 * RemoteControl.
 */
interface RemoteControl {
    void channelUp();
    void channelDown();
    void volumeUp();
    void volumeDown();
}

/**
 * PersonalDevice.
 */
public class PersonalDevice implements RemoteControl {

}
```

3. This program won't compile. Why not?

4. The following code will not compile. Why not?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Father.
 */
class Father {
    /**
     * Talks.
     */
    public final void talk(){
        System.out.println("Manners");
    }
}

/**
 * Son.
 */
public class Son extends Father {
    /**
     * Talks.
     */
    public final void talk(){
        System.out.println("Sloppy");
        super.talk();
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args){
        Son s = new Son();
        s.talk();
    }
}
```

5. This program does not produce the desired output. What modification(s) can be made?

Desired Output:

```
Walk like a Feline
Walk like a Tiger
Walk like a RussianBlue
```

```java
package ca.bcit.comp2522.midterm.prep;

/**
 * Feline.
 */
class Feline {
    public  void walk(){
        System.out.println("Walk like a Feline");
    }
}

/**
 * Tiger.
 */
class Tiger extends Feline {
    public void walk(){
        System.out.println("Walk like a Tiger");
    }
}

/**
 * RussianBlue.
 */
public class RussianBlue extends Tiger {
    /**
     * Walks.
     */
    public void walk(){
        System.out.println("Walk like a RussianBlue");
    }
    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args){
        Feline katya = new RussianBlue();
        katya.walk();
    }
}
```

6. This program does not produce the desired output. There is a compiler error on the specified line. What modification(s) can be made?

Desired Output:
Hello
Hi

```
package ca.bcit.comp2522.midterm.prep.friendly;

/**
 * Me.
 */
public class Me {
    /**
     * Greets.
     */
    void greet() {
        System.out.println("Hello");
    }

}
```

```
package ca.bcit.comp2522.midterm.prep.unfriendly;

import ca.bcit.comp2522.midterm.prep.friendly.Me;

/**
 * Friend.
 */
public class Friend extends Me {
    /**
     * Talks.
     */
    protected void talk(){
        // greet(); ← Compiler error when uncommented!
        System.out.println("Hi");
    }

    /**
     * Drives the program.
     */
    public static void main(String[] args){
        Friend f = new Friend();
        f.talk();
    }
}
```

7. This program does not compile or produce the desired output. Why not? How can you fix it?

Desired Output:
Time is up!

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Wake.
 *
 * @author BCIT
 * @version 2016
 */
public class Wake {

    /**
     * Constructs an object of type Wake.
     *
     * @param time
     */
    public void Wake(String time) {
        System.out.println("Time is " + time);
    }

    /**
     * Drives the program.
     *
     * @param args
     */
    public static void main(String[] args) {
        Wake a = new Wake("up!");
    }
}
```

8. The following code will not compile. Why not?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Duck.
 */
class Duck {

    /**
     * Quacks.
     * @param x
     */
    public void quack(int x){
        System.out.println(x + 5);
    }
}

/**
 * Mallard.
 */
public class Mallard extends Duck {

    /**
     * Quacks.
     * @param x
     */
    public void quack(double x){
        System.out.println(x + 5);
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args){
        Duck ren = new Mallard();
        ren.quack(3.14);
    }
}
```

9. The following code will not compile. Why not?

```
package ca.bcit.comp2522.midterm.prep;

interface Holiday {}

/**
 * Scary.
 */
class Scary implements Holiday {
    public void boo(){
        System.out.println("Boo!");
    }
}

/**
 * Spooky.
 */
class Spooky extends Scary {
    public void boo(){
        System.out.println("Hoo");
    }
}

/**
 * Halloween.
 */
public class Halloween {

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args){
        foo(new Spooky());
    }

    /**
     * @param x
     */
    private static void foo(Holiday x){
        x.boo();
    }
}
```

10. The following code will not compile. Why not?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * USSVoyager.
 */
public class USSVoyager {

    private String captain;

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        if (args[0] != null)
            captain = args[0];
    }
}
```

11. The following code will not compile. Why not?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Phone.
 */
interface Phone {
    void dialNumber();
    void hangUp();
}

/**
 * MobilePhone.
 */
abstract class MobilePhone implements Phone {
    public void dialNumber(int number){
        System.out.println("You are phoning " + number);
    }
    public void hangUp(){
        System.out.println("Hanging up");
    }
}
```

```java
/**
 * SmartPhone.
 */
public class SmartPhone extends MobilePhone {

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args){
        SmartPhone exec = new SmartPhone();
        exec.dialNumber(5551212);
        exec.hangUp();
    }
}
```

12. The following code generates a compiler error on the specified line. Why?

```java
package ca.bcit.comp2522.midterm.prep.pre;

/**
 * Parent.
 */
public class Parent {
    int x;
    protected double y;
}
```

```java
package ca.bcit.comp2522.midterm.prep.post;

import ca.bcit.comp2522.midterm.prep.pre.Parent;

/**
 * Child.
 */
public class Child extends Parent {

    /**
     * Constructs an object of type Child.
     * @param x
     * @param y
     */
    public Child(int x, int y) {
        this.x = x; /* COMPILER ERROR! */
        this.y = y;
```

```
    }
}
```

13. What is the output?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Student.
 */
public class Student {

    int numStudents = 0;

    /**
     * Constructs an object of type Student.
     */
    public Student(){
        numStudents++;
    }

    /**
     * Drives the program
     * @param args
     */
    public static void main(String[] args){
        Student s1, s2, s3;
        s1= new Student();
        s2= new Student();
        s3= new Student();
        System.out.println("Total number of students is " + s3.numStudents);
    }
}
```

14. What is the output if we make the following modification?

```
int numStudents = 0;
  to
static int numStudents = 0;
```

15. Will the program compile if we make the following modification?

```
int numStudents = 0;
```

to

```
static final int numStudents = 0;
```

16. What is the output generated by the following code?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Cup.
 */
class Cup {

    /**
     * Constructs an object of type Cup.
     * @param marker
     */
    public Cup(int marker) {
        System.out.println("Constructing Cup(" + marker + ")");
    }

    /**
     * Sips the amount.
     * @param marker
     */
    void sip(int amount) {
        System.out.println("sipping(" + amount + ")");
    }
}

/**
 * SilverTray.
 */
class SilverTray {
    static Cup firstCup;
    static Cup secondCup;

    static {
        firstCup = new Cup(1);
        secondCup = new Cup(2);
    }

    /**
     * Constructs an object of type SilverTray.
     * @param guest
     */
    SilverTray(String guest) {
        System.out.println("Constructing a SilverTray for " + guest);
```

```
    }
}

/**
 * SilverPlatter.
 */
public class SilverPlatter {

    static SilverTray elizabeth = new SilverTray("elizabeth");
    static SilverTray phillip = new SilverTray("phillip");

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("Executing the main method");
    }

}
```

17. What is the output generated by the following code?

```
package ca.bcit.comp2522.midterm.prep;
/**
 * Shape.
 */
abstract class Shape {
    /**
     * Constructor for objects of type Shape.
     */
    public Shape() {
        System.out.println("Instantiating shape");
        draw();
        System.out.println("Finished drawing shape");
    }
    /**
     * Draws the Shape.
     */
    void draw() {
        System.out.println("Drawing shape");
    }
}
/**
 * Circle.
 */
public class Circle extends Shape {
    private int radius = 1;
    /**
     * Constructs object of type Circle.
     * @param circleRadius
     */
    Circle(int circleRadius) {
        radius = circleRadius;
        System.out.println("Circle has radius = " + radius);
    }
    /**
     * Draws the circle.
     */
    void draw() {
        System.out.println("Drawing  Circle radius = " + radius);
    }
    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        Circle orbit = new Circle(5);
    }
}
```

18. What is the output generated by the following code?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * Greeting.
 */
interface Greeting {
    /**
     * Sends greeting.
     * @return greeting as a String
     */
    String sendGreeting();
}

/**
 * GetWellSoon.
 */
class GetWellSoon implements Greeting {
    /**
     * Sends greeting.
     * @return getWellSoon as a String
     */
    public String sendGreeting(){
        return "GetWellSoon";
    }
}

/**
 * HappyBirthday.
 */

class HappyBirthday implements Greeting {
    public String sendGreeting(){
        return "Bonne fê-ê-ê-te a toi!";
    }
}

/**
 * Hallmark.
*/
public class Hallmark {

    static Greeting cardOne, cardTwo;

    /**
     * Swaps the two object references.
```

```
     * @param a
     * @param b
     */
    public static void swap(Object a, Object b){
        Object temp = a;
        a = b;
        b = temp;
    }
    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args){
        cardOne = new GetWellSoon();
        cardTwo = new HappyBirthday();
        System.out.println(cardOne.sendGreeting());
        swap(cardOne, cardTwo);
        System.out.println(cardOne.sendGreeting());
    }
}
```

19. Given the following code snippet, which of the following are accessible by class B, “x”, “y”, “z”?

```
package ca.bcit.comp2522.midterm.prep.a;

/**
 * A.
 */
public class A {
    private int x;
    int y;
    protected int z;
}
```

```
package ca.bcit.comp2522.midterm.prep.b;

import ca.bcit.comp2522.midterm.prep.a.A;

/**
 * B.
 */
public class B extends A {
    int myX = x;
    int myY = y;
    int myZ = z;
}
```

```java
package ca.bcit.comp2522.midterm.prep.a;

/**
 * C.
 */
public class C {
    A a;

    int myX = a.x;
    int myY = a.y;
    int myZ = a.z;
}

/**
 * D.
 */
class D extends A {
    int myX = x;
    int myY = y;
    int myZ = z;
}
```

20. Which of the following are accessible by class C, “x”, “y”, “z”?

21. What is the output produced by this program?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * MoreExceptionFun.
 */
public class CoolException {

    void f() throws Exception {
        throw new Exception();
    }

    void foo() throws Exception {
        System.out.println("starting foo");
        try {
            System.out.println("in try before f()");
            f();
            System.out.println("in try after f()");
        }
        // catch(Exception e){
        // System.out.println("exception occurred");
        // throw e;
        // }
        finally {
            System.out.println("finally");
        }
        System.out.println("after finally");
    }

    public static void main(String[] args) throws Exception {
        CoolException c = new CoolException();
        try {
            c.foo();
        } finally {
            System.out.println("main finally");
        }
    }
}
```

22. What is the output produced by the previous program if the commented statements are uncommented?

23. What is the output produced by this program?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * ExceptionFun.
 */
public class ExceptionFun {

    void f() throws Exception {
        throw new Exception();
    }

    void foo() throws Exception {
        System.out.println("starting foo");
        try {
            System.out.println("in try before f()");
            f();
            System.out.println("in try after f()");
        } catch (Exception e) {
            System.out.println("exception occurred");
            throw e;
        } finally {
            System.out.println("finally");

        }
        System.out.println("after finally");
    }

    public static void main(String[] args) throws Exception {
        ExceptionFun c = new ExceptionFun();
        try {
            c.foo();
        } finally {
            System.out.println("main finally");
        }
    }
}
```

24. What is the output produced by this program?

```
package ca.bcit.comp2522.midterm.prep;

/**
 * ExceptionTester.
 */
public class ExceptionTester {

    /**
     * Constructor for objects of type ExceptionTester.
     */
    public ExceptionTester() {
        // TODO Auto-generated constructor stub
    }

    public void firstMethod() {
        try {
            System.out.println("starting firstMethod");
            System.out.println("calling secondMethod");
            secondMethod();
            System.out.println("returned from secondMethod");
        } catch (Exception e) {
            System.out.println("firstMethod has caught an exception");
        } finally {
            System.out.println("firstmethod finally!");
        }
    }

    public void secondMethod() {
        try {
            System.out.println("starting secondMethod");
            System.out.println("calling thirdMethod");
            thirdMethod();
            System.out.println("returned from thirdMethod");
        } catch (Exception e) {
            System.out.println("secondMethod has caught an exception");
        } finally {
            System.out.println("secondMethod finally!");
        }
    }

    public void thirdMethod() throws NotYetImplementedException {
        System.out.println("starting thirdMethod");
        throw new NotYetImplementedException();
    }
```

```
    public static void main(String[] args) {
        ExceptionTester tester = new ExceptionTester();
        tester.firstMethod();
    }
}
```

25. What is the OUTPUT produced by this program (continued on next page)?

```
package ca.bcit.comp2522.midterm.prepTwo;

/**
 * Animal.
 */
interface Animal {
    /**
     * Draws.
     */
    void draw();
}

/**
 * Dog.
 */
class Dog implements Animal {
    /**
     * Draws.
     */
    public void draw() {
        System.out.println("Dog");
    }
}

/**
 * Kromfohrlander is a kind of Dog.
 */
class Kromfohrlander extends Dog {
    /**
     * Draws.
     */
    public void draw() {
        super.draw();
        System.out.println("Kromfohrlander");
    }
}

/**
 * Mastiff is a kind of Dog.
```

```
 */
class Mastiff extends Dog implements Animal {
    /**
     * Draws.
     */
    public void draw() {
        System.out.println("Mastiff");
    }
}

class ShibaInu extends Kromfohrlander {
    Mastiff m = new Mastiff();

    /**
     * Draws.
     */
    public void draw() {
        m.draw();
        super.draw();
        System.out.println("ShibaInu");
    }
}

/**
 * Kennel.
 */
public class Kennel {
    /**
     * Drives the program.
     *
     * @param args
     */
    public static void main(String[] args) {
        Animal sasha;
        Dog rex;
        rex = new Mastiff();
        sasha = new ShibaInu();
        sasha.draw();
        rex.draw();
        sasha = rex;
        sasha.draw();
    }
}
```

26. What is the OUTPUT produced by this program?

```
package ca.bcit.comp2522.midterm.prepTwo;

/**
 * MindBender.
 */
public class MindBender {
    /**
     * Gimbles.
     * @param v
     * @throws Exception
     */
    void gimble(int v) throws Exception {
        if (v == 0)
            throw new Exception();
    }

    /**
     * Turlywoopses.
     * @param v
     * @throws Exception
     */
    void turlywoops(int v) throws Exception {
        System.out.println("starting turlywoops");
        try {
            System.out.println("calling foo");
            gimble(v);
            System.out.println("after foo");
        } catch (Exception e) {
            System.out.println("handling exception");
            throw e;
        } finally {
            System.out.println("finally");
        }
        System.out.println("ending function turlywoops");
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        MindBender mindBender = new MindBender();
        try {
            System.out.println("main");
            mindBender.turlywoops(0);
```

```
            System.out.println("after turlywoops");
        } catch (Exception e) {
            System.out.println("main - catch");
        } finally {
            System.out.println("main - finally");
        }
        System.out.println("finished program");
    }
}
```

27. What is the OUTPUT produced by this program?

```
package ca.bcit.comp2522.midterm.prepTwo;

class Alphabet {
    public void display() { System.out.print("Alpha"); }
}

class A extends Alphabet {
    public void display() { System.out.print("A"); }
}

class B extends Alphabet {
    public void display() { System.out.print("B"); }
}

class C extends Alphabet {
    public void display() { System.out.print("C"); }
}

class D extends Alphabet {
    public void display() { System.out.print("D"); }
}

class E extends Alphabet {
    public void display() { System.out.print("E"); }
}

/**
 * Soup.
 */
public class Soup {
    private Alphabet[] bowl;

    /**
     * Makes some soup.
     */
    public void simmer() {
```

```
        bowl = new Alphabet[5];
        bowl[0] = new E();
        bowl[1] = new C();
        bowl[2] = new D();
        bowl[3] = new B();
        bowl[4] = new A();
        for (int i = 0; i < bowl.length; i++) {
            bowl[i].display();
        }
        System.out.println("");

        change(bowl[0], new A());
        bowl[0].display();
        System.out.println("");

        bowl[0] = new E();
        change(bowl, new A(), 0);
        bowl[0].display();
    }

    /**
     * Sets the first parameter to be the second parameter.
     * @param a an Alphabet
     * @param b an Alphabet
     */
    public void change(Alphabet a, Alphabet b) {
        a = b;
    }

    /**
     * Sets the specified index of the specified array to be
     * the specified element.
     * @param a an Alphabet array
     * @param b an Alphabet
     * @param index an int
     */
    public void change(Alphabet[] a, Alphabet b, int index) {
        a[index] = b;
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        Soup pho = new Soup();
        pho.simmer();
    }
```

```
}
```

28. Explain WHY there is an error at the indicated position and PROVIDE A SOLUTION.

```
package ca.bcit.comp2522.midterm.prepTwo;

import java.awt.Point;

/**
 * PathFinder
 */
class PathFinder {
    protected Point origin;

    /**
     * Constructs a PathFinder.
     * @param firstPoint a Point
     */
    public PathFinder(Point firstPoint) {
        this.origin = firstPoint;
    }

    /**
     * Finds the specified Point.
     * @param soughtPoint
     */
    public void find(Point soughtPoint) {
    }
}

/**
 * Ranger.
 */
public class Ranger extends PathFinder {
    Point terminus;

    /**
     * Constructs a Ranger
     *
     * @param origin
     * @param terminus
     */
    public Ranger(Point origin, Point terminus) {// ERROR
        this.origin = origin;
        this.terminus = terminus;
    }

    /**
```

```
     * Finds the specified path.
     *
     * @param origin
     * @param terminus
     */
    public void find(Point origin, Point terminus) {
    }

}
```

29. What is the OUTPUT produced by this program?

```
package ca.bcit.comp2522.midterm.prepTwo;

/**
 * Generalist.
 */
class Generalist {
    static int x = print("x initialized in Generalist");
    static {
        System.out.println("static Generalist");
    }

    /**
     * Prints the specified String and returns an int.
     * @param s
     * @return answer to life the universe and everything
     */
    public static int print(String s) {
        System.out.println(s);
        return 42;
    }

    /**
     * Constructs an object of type Specialist.
     */
    public Generalist() {
        System.out.println("Generalist()");
    }
}

/**
 * Specialist.
 */
public class Specialist extends Generalist {

    static {
        System.out.println("static Specialist");
```

```
    }

    /**
     * Constructs an object of type Specialist.
     */
    public Specialist() {
        System.out.println("Specialist()");
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("starting program");
        new Generalist();
        System.out.println("------------");
        new Generalist();
        System.out.println("------------");
        new Specialist();
        System.out.println("-----------");
        new Specialist();
    }
}
```

30. What is the OUTPUT produced by this program?

```
package ca.bcit.comp2522.midterm.prepTwo;

/**
 * Generalist2.
 */
class Generalist2 {
    static int x = print("x initialized in Generalist2");
    static {
        System.out.println("static Generalist2");
    }

    /**
     * Prints the specified String and returns an int.
     * @param s
     * @return answer to life the universe and everything
     */
    public static int print(String s) {
        System.out.println(s);
        return 42;
    }
```

```
    /**
     * Constructs an object of type Generalist2.
     */
    public Generalist2() {
        System.out.println("Generalist2()");
    }
}

/**
 * Specialist2.
 */
class Specialist2 extends Generalist2 {

    static {
        System.out.println("static Specialist2");
    }

    /**
     * Constructs an object of type Specialist.
     */
    public Specialist2() {
        System.out.println("Specialist2()");
    }
}

/**
 * SpecialistDriver.
 */
public class SpecialistDriver {

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("starting program");
        new Generalist2();
        System.out.println("------------");
        new Generalist2();
        System.out.println("------------");
        new Specialist2();
        System.out.println("-----------");
        new Specialist2();
    }
}
```

31. Why is the OUTPUT in the two previous questions DIFFERENT?

32. What is the OUTPUT produced by this program?

```
package ca.bcit.comp2522.midterm.prepTwo;

/**
 * RecursionGames.
 */
public class RecursionGames {

    private static final int UPPER_BOUND = 5;

    private RecursionGames() { }

    /**
     * Combobulates.
     * @param operand
     */
    public static void combobulate(int operand){
        System.out.println(operand);
        if (operand < UPPER_BOUND) {
            combobulate(operand+1);
        }
        System.out.println(operand);
    }

    /**
     * Drives the program.
     * @param args
     */
    public static void main(String[] args) {
        combobulate(1);
    }
}
```

33. WHY is there a compile-time error in this code snippet? What is a possible SOLUTION?

```
package ca.bcit.comp2522.midterm.prepTwo;

/**
 * Access.
 */
class Access {
    protected void change(){}
}

/**
 * Modifier.
 */
public class Modifier extends Access {
    private void change(){ //ERROR
        System.out.println("changed");
    }
}
```

THE END