Mock In-class Test - 5COSC019W Object Oriented Programming
Duration: 2 Hours
Solutions

PART A - Multiple Choice - 60%

Only one correct answer per question. No negative marking. All questions carry the same weight, 4 marks.
Formatting of this document is not important as the actual test will take place in Blackboard.

Question 1

What happens if a program consisting of a single class does not define a default constructor for that class?

A: The program will not compile.

B: The program will compile but not run and throw an exception.

C: The compiler will sunthesise the default constructor for the class if no other constructors are explicitly defined.

D: You need to define the default constructor of the class in a separate file.

E: The program will normally run, but occasionally crash due to race conditions.

Correct answer: C

Question 2

Which of the following is NOT a characteristic of object oriented programming?

A: Encapsulation

B: Polymath

C: Abstraction

D: Inheritance

Correct answer: B

Question 3

What is the output of the following code?

```
class E {
    private int a = 10;
    private int b;

    public static void main(String[] args) {
        E x = new E();
        E y = new E();
        E z = new E();
        x.a = y.b;
        x = z;
        z = x;
        y.a = 12;
        z.a = z.a + x.a;

        x = new E();
        x.a = 30;
        x.b = 21;
        y.b = 22;
        System.out.println(x.a + " " + y.a + " " + z.a + " " + x.b +
                           " " + y.b);
    }
}
```

A: 30 12 30 21 22

B: 30 12 20 21 22

C: 30 30 21 21 22

D: 30 12 60 21 22

E: 60 12 60 21 22

Correct answer: B

Question 4

What is the output of the following code?

```
class References3 {
    int a = 1;
    int b = 8;

    public static void main(String[] args) {
        References3 r1 = new References3();
        r1.b = 10;
        References3 r2 = new References3();
```

```
        r2.b = r1.a++;

        r1 = r2;
        r2.b++;

        References3 r3 = new References3();
        r1 = r3;
        System.out.println(r1.a + " " + r1.b + " " + r2.a + " " + r2.b);
    }
}
```

A: 1 8 1 3

B: 1 8 2 2

C: 1 8 2 3

D: 1 2 1 8

E: 1 8 1 2

Correct answer: E

Question 5

What is the output of the following segment of code?

```
class MyNumber2 {
    int x = 5;

    MyNumber2(int x) {
        this.x = x;
    }
}

class M {
    void increase(int i) {
        ++i;
    }
    void increase(MyNumber2 m) {
        m.x++;
    }
    public static void main(String[] args) {
        int x = 19;
        M m = new M();
        m.increase(x);
        MyNumber2 n = new MyNumber2(5);
        m.increase(n);
        System.out.println(x + " " + n.x);
    }
}
```

A: 19 20

B: 19 5

C: 20 5

D: 19 6

E: 20 6

Correct answer: D

Question 6

What is the output of the following program?

```
abstract class A {
    void foo() {
        System.out.print("A ");
    }
}

class B extends A {
    void foo() {
        System.out.print("B ");
    }
}

class C extends B {
    void foo() {
        System.out.print("C ");
    }

    public static void main(String[] args) {
        A a1 = new C();
        a1.foo();
        B a2 = new B();
        a2.foo();

        A a3 = a1;
        a3.foo();
    }
}
```

A: A B C

B: A B B

C: A B A

D: C B A

E: C B C

Correct answer: E

Question 7

What is the output of the following program?

```
class Q7 {
    int a;
    Q7(int b) {
        a = b;
    }

    public static void main(String[] arg) {
        Q7 q1 = new Q7(5);
        Q7 q2 = new Q7(5);

        if (q1.equals(q2))
            System.out.println(true);
        else
            System.out.println(false);

        String s1 = new String("ab");
        String s2 = new String("ab");

        if (!s1.equals(s2))
            System.out.println(false);
        else
            System.out.println(true);
    }
}
```

A: `true true`

B: `false false`

C: `false true`

D: `true false`

E: The program will throw an exception during its execution.

Correct answer: C

Question 8

Which of the following statements is the most accurate about the program below?

```
class X {
    void bar() {}
    void bar(int i) {}
}

class Y extends X {
```

```
    void bar() {}
}

class Z {
    void bar() {}
}
```

A: method `bar` is overridden in X and also overloaded in X.

B: method `bar` is overridden in Y and also overloaded in X.

C: method `bar` is overloaded in Y and also overridden in X.

D: method `bar` is overridden in Y and Z and also overloaded in X.

E: method `bar` is overloaded in X, Y and Z.

Correct answer: B

Question 9

What is the output of the following segment of code?

```
class Q9 {
     public static void main(String args[]) {
        String string = "DOG";

        if (string == "DOG")
            System.out.print("Equal ");
        else
            System.out.print("Not Equal ");


        if ("DOG".equals("DOG"))
            System.out.print("Equal ");
        else
            System.out.print("Not Equal ");

        if ("DOG" == new String("DOG"))
            System.out.print("Equal ");
        else
            System.out.print("Not Equal ");

    }
}
```

A: `Equal, Equal, Equal`

B: `Not Equal, Equal, Not Equal`

C: `Not Equal, Equal, Equal`

D: `Equal, Equal, Not Equal`

E: `Not Equal, Not Equal, Equal`

Correct answer: D

Question 10

What is the output of the following code?

```
class A {
    int x = 5;

    A(int x) {
        this.x = x;
    }

    void foo() {
        System.out.print("A ");
    }
}


class B extends A {
    int x;

    B(int x) {
        super(x);
    }

    void foo(){
        System.out.print(super.x + " " + x + " ");
        super.foo();
    }
}


class Q10 {
    public static void main(String[] args) {
        B b1 = new B(10);
        b1.foo();
    }
}
```

A: 10 0 A

B: 10 10 A

C: 5 10

D: 5 10 A

E: 5 0 A

Correct answer: A

Question 11

What is the output of the following code?

```
public class Q11 {
    public static void main(String[] args) {
        String x[] = new String[6];
        x[0] = "10";
        x[2] = "-1";

        String y[] = new String[2];
        y = x;  // line 9
        y[2] = "abc";
        y[5] = "123";

        System.out.print(x[5] + " " + y[5]);
    }
}
```

A: The code would not even compile. There are no 5 elements in y

B: 0 123

C: 123 123

D: 0 0

E: 123 0

F: The code will compile but it will throw an exception during running it because you cannot assign an array to another array if they have different lengths (line 9)

Correct answer: C

Question 12

What is the output of the following segment of code?

```
class Pen {
    static int i = 0;
    int c = 0;

    Pen(int x) {
        i++;
        c = x;
```

```
    }
}

class Q12 {
    public static void main(String[] a) {
        Pen p1 = new Pen(5);
        Pen p2 = new Pen(7);

        System.out.println(p1.i + " " + p1.c + " " + p2.i + " " + p2.c);
    }
}
```

A: 1 5 1 7

B: 1 7 1 7

C: 2 7 2 7

D: 1 5 2 7

E: 2 5 2 7

Correct answer: E

Question 13

Which of the following statements for Java access specifiers is the most accurate?

A: A *private* member can be accessed only by the class itself. A *protected* member can be accessed by subclasses and classes of the same package. The default access is access by all the classes in the same package.

B: A *private* member can be accessed only by the class itself. A *protected* member can be accessed by subclasses. The default access is access by all the classes in the same package.

C: A *private* member can be accessed only by the class itself. A *protected* member can be accessed by classes of the same package. The default access is access by all the classes in the same package.

D: A *private* member can be accessed only by the class itself. A *protected* member can be accessed by subclasses and classes of the same package. The default access is access by all the classes in the same file.

E: A *private* member can be accessed only by the class itself and its subclasses. A *protected* member can be accessed by subclasses and classes of the same package. The default access is access by all the classes in the same package.

Correct answer: A

Question 14

What is the output of the following segment of code?

```
class A1 {
    A1() {
        System.out.println("A1a");
    }

    A1(int x) {
        System.out.println("A1b");
    }
}

class A2 extends A1 {
    A2() {
        super(12);
        System.out.println("A2");
    }
}

class A3 extends A2 {
    A3() {
        System.out.println("A3");
    }
}


class Q14 {
    public static void main(String[] x) {
        A1 a = new A3();
    }
}
```

A: A1a A2 A3

B: A1b A2 A3

C: A3

D: A1

E: A1a A1b A2 A3

F: The program will not compile! You cannot assign an A3 object to an A1 reference variable

G: The program will not compile! There is `args` in the main method.

Correct answer: B

Question 15

What will happen when you attempt to compile and run the following code?

```
public class Background extends Thread {
    public static void main(String argv[]) {
        Background b = new Background();
        b.run();
    }

    public void start() {
        for (int i = 0; i < 10; i++) {
            System.out.println("Value of i = " + i);
        }
    }
}
```

A: A compile time error indicating that no run method is defined for the Thread class.

B: A run time error indicating that no run method is defined for the Thread class.

C: The code compiles and at run time the values 0 to 9 are printed out.

D: The code compiles but there is no output at runtime.

Correct answer: D

PART B - Programming - 40%

Implement the Java classes (with full details of all fields and methods, including constructors and getter, setter methods) which are required to simulate the following problem:

A hotel is located in a city and it has 50 rooms that visitors can book. A room is assigned to a visitor by the hotel. Every time that a visitor checks-in the hotel they are assigned to a room randomly.

Every time that a visitor checks out of the hotel, the room they were stayingbecomes empty.

A room has an occupant (visitor, thefore the room is not available), or it might be empty (available) and also it is located in a specific floor of the hotel.

Each visitor has a name, address and has a favourite hotel with a specific name.

As part of your implementation, in addition to the classes described in the scenario above, you should also provide a `Simulation` class in which a visitor checks-in a hotel and checks out for 20 times.

You do not need to include getters and setters for every single field. Assuming that you implement the constructor of a class to initialise the fields, you could just provide getters and setters for 1 or 2 fields only).

Marking Scheme: Correct identification of classes `Hotel`, `Room`, `Visitor` 6 marks (2 each). Correct implementation of fields and constructors of these classes 9 marks (3 marks each). Correct implementation of getter/setter methods: 3 marks (1 for each class). Total: 18 marks.

Correct implementation of `check-in` methods with probabilities 10 marks. Correct implementation of `check-out` method: 6 marks.

Implementation of `Simulation` class: 6 marks.

Sample code:

```
// Sample implementation - this could be different
import java.util.Random;

class Hotel {
    Room[] rooms = new Room[50];
    String city;
    Random gen = new Random();
    int num_occupied = 0;  // how many rooms are occupied

    Hotel() {
        for (int i=0; i < 50; i++)
            rooms[i] = new Room(i, gen.nextInt(10), null);
    }

    Hotel(String city) {
        this.city = city;
for (int i=0; i < 50; i++)
            rooms[i] = new Room(i, gen.nextInt(10), null);
    }

    String getCity() {
        return city;
    }

    void setCity(String city) {
        this.city = city;
    }

    void checkIn(Visitor c) {
if (num_occupied < 50) {
    int room_selected = gen.nextInt(50);

    while (rooms[room_selected].getVisitor() != null) {
room_selected = gen.nextInt(50);
    }

    rooms[room_selected].setVisitor(c);
    ++num_occupied;

    System.out.println(c.getName() + " checks-in room " + room_selected);
}
else
    System.out.println("All rooms are full!");
    }
```

```
    void checkOut(Visitor c) {
--num_occupied;
        for (Room r: rooms) {
            if (r.getVisitor() == c) {
r.setVisitor(null);

System.out.println(c.getName() + " checks-out from room " + r.number);
    }
}
    }
}

class Visitor {
    String name;
    String address;
    String favouriteHotel;

    Visitor() {

    }

    Visitor(String name, String address, String favouriteHotel) {
        this.name = name;
        this.address = address;
        this.favouriteHotel = favouriteHotel;
    }

    String getName() {
        return name;
    }

    void setName(String name) {
        this.name = name;
    }
}

class Room {
    int number;
    int floor;
    Visitor visitor;

    Room(int number, int floor, Visitor occupant) {
        this.number = number;
this.floor = floor;
this.visitor = occupant;
    }

    Visitor getVisitor() {
        return visitor;
    }
```

```
    void setVisitor(Visitor v) {
        visitor = v;
    }


}

class Simulation {
    public static void main(String[] args) {
        Hotel hotel = new Hotel("Paris");
        Visitor p = new Visitor("Joe", "Kensington Street", "LocalHotel");

        for (int i=1; i<=20; i++) {
            hotel.checkIn(p);
            hotel.checkOut(p);
        }
    }
}
```