Programming Language Concepts

Mid-Semester Examination, II Semester 2022-2023

Date : 21 February, 2023
Time : 0930 – 1230

Marks : 30
Weightage : 20%

- *Answer all questions.*
- *In any question that asks for code, provide Java-like pseudocode. Syntax errors will be ignored, provided the code is conceptually correct.*
- *Supply explanations for any code you write, ideally as annotations alongside the code.*

---

1. Assume you have a `List` class defined on the right. You wish to add the following functionality to loop through all elements in the list.

```
List myList = new List(100);
... add elements to myList ...

for (int elem: myList) {
    ...
}
```

Modify the class `List`, while still keeping the variables `elems` and `length` private, to allow such a loop, and explain how to express the `for` loop in terms of your modified class definition.

```
class List {
    private int[] elems;
    private int length;

    List(int n) {
        length = n;
        elems = new int[n];
    }

    boolean add(int n) {
        ...
    }
}
```

(4 marks)

2. Suppose we have a `Shape` class, and several subclasses like `Circle`, `Rectangle`, `Triangle`, `Square` etc. Assume also that we have generic `List` class.

Explain the difference between the following two declarations:

```
List<Shape> myShapes1 = new List<Shape>();
List<Shape> myShapes2 = new List<T extends Shape>();
```

(2 marks)

3. Consider the following class definitions in Java:

```
public abstract class Animal {
    public abstract int limbs();
    public boolean equals(Animal a) {
        return (this.limbs() == a.limbs());
    }
}
```

```
public class Bird extends Animal {
    private boolean flies;
    public Bird(boolean canfly) { flies = canfly; }
    public int limbs () {
        if (flies) { return 4;} else { return 2;}
    }
    public boolean equals(Bird a) {
        return (this.limbs() == a.limbs());
    }
}
```

Mammals

```
public class Bird extends Animal {
    private boolean flies; swims;
    public Bird(boolean canfly) { flies = canfly; }
    public int limbs () {
        if (flies) {return 4;} else {return 2;}
    }
    public boolean equals(Bird a) {
        return (this.limbs() == a.limbs());
    }
}
```

Describe and explain the output produced by the following code:

```
public class TestAnimals{
    public static void main(String[] args){
        Bird ostrich = new Bird(false);
        Mammal dolphin = new Mammal(true);
        Object d = dolphin;
        Animal o = ostrich;

        System.out.println("1. " + d.equals(ostrich));
        System.out.println("2. " + d.equals(dolphin));
        System.out.println("3. " + o.equals(ostrich));
        System.out.println("4. " + o.equals(dolphin));
        System.out.println("5. " + ostrich.equals(d));
        System.out.println("6. " + dolphin.equals(o));
    }
}
```

(4 marks)

4. Consider a function `flatten()` that takes as input a list of lists of arbitrary depth and flattens it to a top level list. For instance:

- `flatten([[1,2],[3,4]])` yields `[1,2,3,4]`
- `flatten([[5],[6],[7],[8]])` yields `[5,6,7,8]`

You can assume that the input list has a homogenous type.

This function cannot be written in Haskell because it is not possible to assign a static type to the function. It can be written in Python by inspecting the types of the values in the input list and flattening them recursively.

Suppose we wish to write such a function in Java that takes an `ArrayList<T>` as argument, where `T` could represent a nested `ArrayList` type. Will it be possible to write this function in Java? Explain your answer. *(8 marks)*

5. Consider the following program.

```
void main() {
    int count = 0;
    int fib(int n) {
        int a, b = 0;
        if (n <= 1) then return(1);
        a = fib(n-1); b = fib(n-2);
        count += 1;
        return(a+b);
    }
    print(fib(4));
    print(count);
}
```

Show the successive contents of the system stack on calling `main` (starting from the empty stack). You should clearly demarcate each stack frame and display the function name and parameters, You should also show the values of all the relevant variables and the control link and access link in each activation record. *(8 marks)*

6. Consider the following fragment of Java code, where we assume that each `Employee` object has two (public) attributes `name` and `salary`.

```
Employee emp1, emp2;
for (i = 0; i < 100; i++) {
    emp1 = new Employee("Name", 100.0);
    emp1 = new Employee(emp1.name, emp1.salary);
    emp2 = new Employee(emp1.name, emp1.salary);
    emp1 = emp2;
}
```

Assuming that each instance of class `Employee` occupies `b` bytes of memory, how much garbage is generated by the above code? *(4 marks)*