1. (2 marks) What are the outputs of the following Scala code segment?

```scala
scala> var a=Array(1,3,5);
scala> var b=Array(2,4,6);
scala> for (e <- a) yield e%2

Answer: Array[Int] = Array(1, 1, 1)
```

```scala
scala> for (i <- a; j <- b; if j%i==0) yield (i,j)

Answer: Array[(Int, Int)] = Array((1,2), (1,4), (1,6), (3,6))
```

2. (10 marks) Consider the following definitions representing trees of integers.

```scala
abstract class IntTree

case class EmptyTree() extends IntTree

case class Node(elem:Int, left:IntTree, right:IntTree) extends IntTree
```

Implement the function `contains` which returns `true` if the given `IntTree` object `t` contains the value `v` and `false` otherwise.

```scala
def contains(t:IntTree, v:Int):Boolean = t match {
  // your code here
}
```

Answer:

```scala
def contains(t: IntTree, v: Int): Boolean = t match {
  case Empty() ⇒ false
  case Node(e, l, r) ⇒ if (e==v) true else contains(l, v) || contains(r, v)
}
```
3. (8 marks) For the following Scala expressions, provide equivalent implementations using higher-order functions.

```scala
var a=Array(1,3,5);
for (e <- a) yield 2*e
```

**Answer:** `a.map(2*_)`

```scala
for (e <- a; if e%2==0) yield e
```

**Answer:** `a.filter(_%2==0)`

4. (8 marks) The following `Exp` defines a Summation expression.

```scala
abstract class Expr {
  case class Number(n:Double) extends Expr
  case class Sum(e1:Expr, e2:Expr) extends Expr
}
```

According to the defined `eval` function, what are the outputs of the Scala code?

```scala
object Q2 {
  def main(args:Array[String]){  
    val e=Sum(Sum(Number(1),Sum(Number(2),Number(3)))), Number(4))
    println(eval(e));
  }
  def eval(e:Expr):Double = e match{
    case Number(n) ⇒ {println(n); n}
    case Sum(e1,e2) ⇒
      {val a=eval(e1); val b=eval(e2); println(e1+", "+e2); a+b}
  }
}
```

**Answer:**

```
1.0
2.0
3.0
Number(2.0), Number(3.0)
Number(1.0), Sum(Number(2.0),Number(3.0))
4.0
Sum(Number(1.0),Sum(Number(2.0),Number(3.0))), Number(4.0)
10.0
```
5. **(12 marks)** Based on the given class hierarchy, state if each of the given Scala statement in the Table is legal or not. If the statement is legal and the statement is a method call, provide the called class::method. If the statement creates a new object, list the class constructors called in correct order.

<table>
<thead>
<tr>
<th>Scala code</th>
<th>legal (Y/N)</th>
<th>class::method/class::constructors called</th>
</tr>
</thead>
<tbody>
<tr>
<td>val a=new A(100)</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>val b=new C(101)</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>val e=new E(102)</td>
<td>Y</td>
<td>constructors A,C,B,E</td>
</tr>
<tr>
<td>e.m1</td>
<td>Y</td>
<td>B:m1</td>
</tr>
<tr>
<td>e.m2</td>
<td>Y</td>
<td>B:m2</td>
</tr>
<tr>
<td>e.m3</td>
<td>Y</td>
<td>E:m3</td>
</tr>
<tr>
<td>e=new E(103)</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>val f=new F(104)</td>
<td>Y</td>
<td>constructors A,B,C,F</td>
</tr>
<tr>
<td>f.m4;</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>var i=new I(107);</td>
<td>Y</td>
<td>constructors A,C,B,D,I</td>
</tr>
<tr>
<td>i.m1</td>
<td>Y</td>
<td>D:m1</td>
</tr>
<tr>
<td>i=new I(108)</td>
<td>Y</td>
<td>constructors A,C,B,D,I</td>
</tr>
</tbody>
</table>
6. **(10 marks)** The objective of the course is to prepare you for programming related job positions by introducing to you different programming paradigms, e.g. imperative, functional, object-oriented and the combination of the three. Among the 4 studied languages, discuss which of them you like the most and which of them you like the least and why. There is no one correct answer. Any opinion you expressed is valid.