

MODULE 5 - SHEET 1

```
public class RecFrac
{ public static void main(String[] args)
  { for (int i=0; i<=10; i++)
    System.out.println(f(i));
  }

  private static double f(int n)
  { if (n == 0)
    return 2.0d;
    else
    return 1.0d + 1.0d/(1.0d+f(n-1));
  }
}

// The body of method f may be written more economically:
//
// { return n==0 ? 2.0d : 1.0d+1.0d/(1.0d+f(n-1));
// }
```

```
public class Factorial
{ public static void main(String[] args)
  { for (int i=0; i<=10; i++)
    System.out.println(fac(i));
  }

  private static int fac(int n)
  { return n==0 ? 1 : n*fac(n-1);
  }
}
```

```
public class Hanoi
{ public static void main(String[] args)
  { move(3, "A", "B", "C");
  }

  private static void move(int n, String p, String q, String r)
  { if (n>0)
    { move(n-1, p, r, q);
      System.out.println("Move disc " + n +
        " from peg " + p + " to peg " + r);
      move(n-1, q, p, r);
    }
  }
}
```

MODULE 5 - SHEET 2

```
public class FactorialWR
{ public static void main(String[] args)
  { System.out.println(fac5());
  }

  private static int fac5()
  { return 5*fac4();
  }

  private static int fac4()
  { return 4*fac3();
  }

  private static int fac3()
  { return 3*fac2();
  }

  private static int fac2()
  { return 2*fac1();
  }

  private static int fac1()
  { return 1*fac0();
  }

  private static int fac0()
  { return 1;
  }
}

public class HanoiWR
{ public static void main(String[] args)
  { move3("A", "B", "C");
  }

  private static void move3(String p, String q, String r)
  { move2(p, r, q);
    System.out.println("Move disc 3" +
                      " from peg " + p + " to peg " + r);
    move2(q, p, r);
  }

  private static void move2(String p, String q, String r)
  { move1(p, r, q);
    System.out.println("Move disc 2" +
                      " from peg " + p + " to peg " + r);
    move1(q, p, r);
  }
}
```

```
private static void move1(String p, String q, String r)
{ move0(p, r, q);
  System.out.println("Move disc 1" +
                    " from peg " + p + " to peg " + r);
  move0(q, p, r);
}

private static void move0(String p, String q, String r)
{}
}
```

MODULE 5 - SHEET 3

```
// The following is a complete Java program. The problem is to analyse
// the program and determine what it writes out WITHOUT keying the program
// in and running it.
```

```
public class SetUp

{ public static void main(String[] args)
  { Child alf = new Child(11);
    System.out.println("alf.H is " + alf.getH());
  }
}

class Child extends Parent
{ private int H;

  public Child(int j)
  { super(j);
    this.setH();
  }

  public void set(int j)
  { super.set(j);
    this.setH();
  }

  private void setH()
  { this.H = 2*this.getK();
    System.out.println("*** I've just set H to " + H + " ***");
  }

  public int getH()
  { return this.H;
  }
}

class Parent
{ private int J, K;

  public Parent(int j)
  { this.set(j);
  }

  protected void set(int j)
  { this.J = j;
    System.out.println("*** I've just set J to " + J + " ***");
    setK();
  }

  private void setK()
  { this.K = 2*J;
    System.out.println("*** I've just set K to " + K + " ***");
  }
}
```

```
    }  
  
    public int getK()  
    { return this.K;  
    }  
}
```

MODULE 5 - SHEET 4

// The following is a complete Java program. The problem is to analyse
 // the program and determine what it writes out WITHOUT keying the program
 // in and running it. Instances of two pairs of square brackets refer
 // to a two-dimensional array. Thus int[4][4] is a 4x4 array. Note
 // that class GPS contains classes within itself. These are known as
 // member classes. The item GPS in GPS.this.i[0] indicates that the
 // relevant this is the one associated with the instantiation of GPS.

```
public class GPSprog
{ public static void main(String[] args)
  { GPS g = new GPS();
    for (int i=0; i<4; i++)
      { for (int j=0; j<4; j++)
        { System.out.print(g.a[i][j] + " ");
          System.out.println();
        }
      }
  }
}

class GPS
{ public int[] [] a = new int[4][4];
  private int[] i = new int[1];
  private int[] j = new int[1];

  public GPS()
  { this.i[0] = this.gps(this.j, 4, new Passi(), new Fevalg());
  }

  private int gps(int[] i, final int N, Pass z, Feval v)
  { i[0] = 0;
    while (i[0]<N)
      { z.p(v.f());
        i[0]++;
      }
    return 0;
  }

  private abstract class Pass
  { public abstract void p(int n);
  }

  private class Passi extends Pass
  { public void p(int k)
    { GPS.this.i[0] = k;
    }
  }

  private class Passaij extends Pass
  { public void p(int k)
    { GPS.this.a[GPS.this.i[0]][GPS.this.j[0]] = k;
    }
  }
}
```

```

    }
}

private abstract class Feval
{ public abstract int f();
}

private class Fevaliplusj extends Feval
{ public int f()
  { return GPS.this.i[0]+GPS.this.j[0];
  }
}

private class Fevalg extends Feval
{ public int f()
  { return GPS.this.gps(GPS.this.i, 4, new Passaij(),
                        new Fevaliplusj());
  }
}
}

```