# Sample Program 12.3

**Using Objects**

**Problem statement**. Modify Exercise 4.3's `MySavingsPlan1.java` by replacing its separate variables with a single object with data fields that replace the variables. Name the new file **`MySavingsPlanWithObjectsAndFunctions.java`**.

Create an object specification named **`class Savings`**, with the data fields for:

1. the amount deposited every month (D in previous versions)

2. interest rate (p in previous versions)

3. number of years to make deposits (years in previous versions)

4. the amount saved (S in previous versions)

Then in your program, declare an *object* of type **`Savings`***,* and use its data fields instead of the four separate variables. Include any additional data fields that you may wish to include beyond these four.

Also create a function named calculate, that takes a Savings object as its parameter, and performs all the calculations for the object. Call the function from main, instead of calculating in main.

**Solution.**

```
import java.io.*;
import java.text.*;

class Savings
{
  int years; // years of savings
  int D;     // dollars deposited every month
  double R;  // annual interest rate, percent
  double p;  // monthly interest rate, decimal, calculated
  double T;  // term of savings plan in months, calculated
  double S;  // total saved with interest, calculated
};

public class MySavingsPlanWithObjectsAndFunctions
{
  static void calculate(Savings s)
  {
    s.p = s.R / 100 / 12;
    s.T = s.years * 12;
    s.S = s.D * ((Math.pow(1 + s.p, s.T) - 1) / s.p);
  }

  public static void main(String[] args)
  {
    // create a savings plan object with initial values
    Savings savings = new Savings();
    savings.years = 10;
    savings.D = 100;
    savings.R = 7.5;

    // output (calculated) values
    calculate(savings);

    // echoing input values, unformatted
    System.out.print("In " + savings.years + " years, $");
    System.out.print(savings.D + " deposited per month will grow to $");

    // formatting output (see 4.2)
    System.out.println(new DecimalFormat("#,000.00").format(savings.S));
  }
}
```