Josephus Simulation with *Mathematica 3.0*

by *Michael A. Rundel*

Preface

I have written this example for a *Mathematica* Demonstration held at the Kuffner Observertory in December 1998. I used it to demonstrate some powerful concepts and features of *Mathematica*, like functional programming, pattern matching, pure functions, functions that remember values, list construction and manipulation and - last but not least - Mathematica's superior Graphic Tools.

The Problem

Josephus Flavius was a famous Jewish historian of the first century at the time of the Second Temple destruction. During the Jewish-Roman war he got trapped in a cave with a group of 40 soldiers surrounded by romans. The legend has it that preferring suicide to capture, the Jews decided to form a circle and, proceeding around it, to kill every third remaining person until no one was left. Josephus, not keen to die, quickly found the safe spot in the circle and thus stayed alive.[1]

The *Mathematica* Simulation

Imagine each Conspirator is identified by a number written on his back, starting from the first one. The whole group of Conspirators thus can be represented by a list of consecutive numbers. This can be generated by the Function `Range[imax]` which yields to the list `{1, 2,..., imax}`. The counting of *n* places to the right is realised by cyclic rotating the list *n* steps to the left, so that the next person going to die becomes the first element in the list. This is achieved by applying the `RotateLeft[]` Function to the list of Conspirators. Now the first element (person) is eliminated, which is done by the `Rest[]` Function. Since in the end there is only one person alive, there have to be *(numberOfConspirators-1)* persons eliminated, so the procedure is applied *(numberOfConspirators-1)* times, which is done by the `Nest[]` Function.

```
In[1]:=
    Josephus[numberOfConspirators_?Positive, steps_?Positive]:=
    Josephus[numberOfConspirators, steps]=
                    Nest[
                    Rest[RotateLeft[#, steps]]&,
                    Range[numberOfConspirators],
                    numberOfConspirators-1]
```

Now we can answer with ease where to survive in a ring of 10 conspirators when every third is about to commit suicide:

```
In[2]:=
    Josephus[40, 3]

Out[2]:=
    {7}
```

so Josephus had to take place seven in order to survive the masaquere. And here is another one:

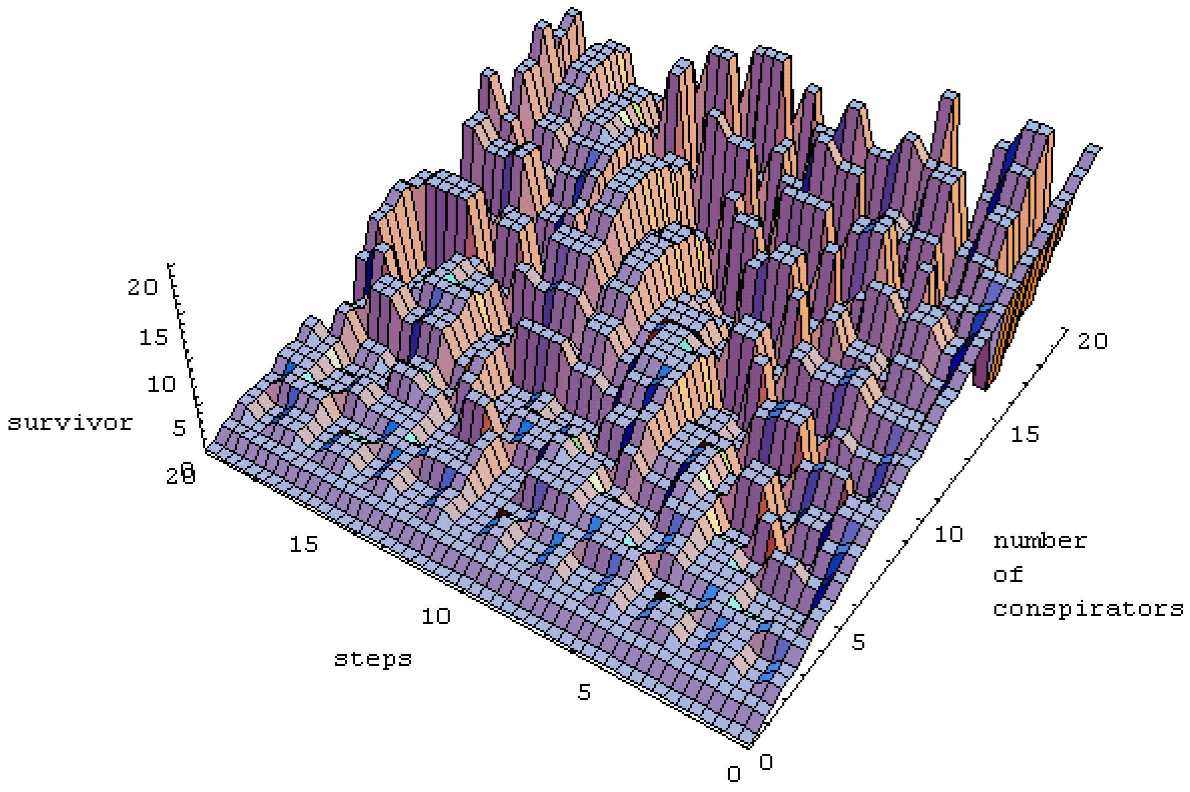
```
In[3]:=
     Josephus[40, 5]

Out[3]:=
     {28}
```

When we now take a look at the definition of the function `Josephus[]`, we see that the already calculated values are stored in an 2-dimensinal array `Josephus[x,y]`.

```
In[4]:=
     ?Josephus

Out[4]:=
     "Global`Josephus"
     Josephus[40, 3] = {7}

     Josephus[40, 5] = {28}

     Josephus[(numberOfConspirators_)?Positive, (steps_)?Positive] :=
           Josephus[numberOfConspirators, steps] =
             Nest[Rest[RotateLeft[#1, steps]] & , Range[numberOfConspirators],
               numberOfConspirators - 1]
```

That the values for `Josephus[]` are stored in an array once calculated saves a lot of calculation time, when the Function is frequently called with the same arguments. This may not only be the case when defining recursive Functions, but also when, for example, plotting the result of a function with discrete arguments.

```
In[5]:=
     Plot3D[
             First[Josephus[Round[x],Round[y]]], {x, 0, 20}, {y, 0, 20},
             {ViewPoint -> {-2.053, -1.372, 2.312},
              PlotPoints -> 50,
              Boxed -> False,
              AxesLabel -> {"number\nof\nconspirators",
                            "steps",
                            "survivor"}}
     ];
```



Now to get a glimpse of Mathematica's power of functional programming suppose you would like to modify the original `Josephus[]` function, so that you can actualy observe the elimination process. This is done by the function `JosephusList[]`, which produces a list of all intermediate steps. Not however, that this can simply be done by replacing the `Nest[]` function with the `NestList[]` function in the original definition of `Josephus[]`.

In[6]:=

```
JosephusList[numberOfConspirators_?Positive, steps_?Positive]:=
        NestList[
                Rest[RotateLeft[#, steps]]&,
                Range[numberOfConspirators],
                numberOfConspirators-1];

JosephusList[10, 2]
```

Out[6]:=

```
{{1,2,3,4,5,6,7,8,9,10},
 {4,5,6,7,8,9,10,1,2},
 {7,8,9,10,1,2,4,5},
 {10,1,2,4,5,7,8},
 {4,5,7,8,10,1},
 {8,10,1,4,5},
 {4,5,8,10},
 {10,4,5},
 {10,4},
 {4}}
```

That's all. Now I have to go back to my own conspiray...;-)

Reference

[1] R.Graham, D.Knuth, O.Potashnik, Concrete Mathematics, 2nd edition, Addison-Wesley, 1994.