In[1]:=
```
<< RISC`GeneratingFunctions`
```

```
Package GeneratingFunctions version 0.8 written by Christian Mallinger
Copyright Research Institute for Symbolic Computation (RISC),
Johannes Kepler University, Linz, Austria
```

In[2]:=
```
? REPlus
```

```
RecurrenceEquationPlus[re1,re2,a[n]] gives a recurrence equation that
is satisfied by the sum of solutions of the recurrences re1 and re2.
All recurrences are given in a[n].

Alias: REPlus

See also: REInfo, DEPlus
```

```
(* defining the sequences of Perrin and Lucas numbers *)
```

In[3]:=
```
perrin = {a[n + 3] - a[n + 1] - a[n] == 0, a[0] == 3, a[1] == 0, a[2] == 2};
lucas = {a[n + 2] - a[n + 1] - a[n] == 0, a[0] == 2, a[1] == 1};
```

```
(* computing a recurrence for the sum of Perrin and Lucas numbers *)
```

In[5]:=
```
REPlus[perrin, lucas, a[n]]
```

Out[5]=
```
{a[n] + 2 a[1 + n] - 2 a[3 + n] - a[4 + n] + a[5 + n] == 0,
 a[0] == 5, a[1] == 1, a[2] == 5, a[3] == 7, a[4] == 9}
```

In[6]:=
```
? RE*
```

▼ RISC`GeneratingFunctions`

RE	RE2L	REHadamard	REInterlace	REPlus	RESubsequence
RE2DE	RECauchy	REInfo	REOut	REShadow	

In[7]:=
```
? RESubsequence
```

```
RecurrenceEquationSubsequence[re,a[n],m*n+k] gives a recurrence that
is satisfied by a subsequence of the form a[m*n+k] of every solution a[n] of
the input recurrence re.

Alias: RESubsequence

See also: REInfo, REInterlace
```

```
(* Computing a recurrence for all odd-indexed Lucas numbers *)
```

In[8]:=
```
RESubsequence[lucas, a[n], 2 n + 1]
```
Out[8]=
```
{a[n] - 3 a[1 + n] + a[2 + n] == 0, a[0] == 1, a[1] == 4}
```

```
(* Cauchy product of Perrin and Lucas numbers *)
```

In[9]:=
```
RECauchy[perrin, lucas, a[n]]
```
Out[9]=
```
{a[n] + 2 a[1 + n] - 2 a[3 + n] - a[4 + n] + a[5 + n] == 0,
 a[0] == 6, a[1] == 3, a[2] == 13, a[3] == 20, a[4] == 34}
```

```
(* defining the Fibonacci sequence *)
```

In[10]:=
```
fib = {F[n + 2] - F[n + 1] - F[n] == 0, F[0] == 0, F[1] == 1};
```

```
(* Computing the recurrence for the Cauchy
 product of Fibonacci numbers with the constant sequence 1,
i.e., for the partial sum of Fibonacci numbers *)
```

In[11]:=
```
RECauchy[fib, {F[n] == 1}, F[n]]
```
Out[11]=
```
{F[n] - 2 F[2 + n] + F[3 + n] == 0, F[0] == 0, F[1] == 1, F[2] == 2}
```

```
(* Perrin and Lucas numbers are built-in *)
```

In[12]:=
```
? Perrin
```

Perrin[n] gives the nth Perrin number.

In[13]:=
```
? LucasL
```

LucasL[n] gives the Lucas number L_n.
LucasL[n, x] gives the Lucas polynomial $L_n(x)$. ≫

```
(* creating data for guessing *)
```

In[14]:=
```
dataL = Table[LucasL[nn], {nn, 0, 20}];
dataP = Table[Perrin[nn], {nn, 0, 20}];
```

In[16]:=
```
? GuessRE
```

```
(* Guessing the recurrence for Lucas numbers *)
```

In[17]:=
```
GuessRE[dataL, a[n]]
```
Out[17]=
```
{{-a[n] - a[1 + n] + a[2 + n] == 0, a[0] == 2, a[1] == 1}, ogf}
```

```
(* Guessing the recurrence for Perrin numbers →
 need to increase the maximal order used! *)
```

In[18]:=
```
GuessRE[dataP, a[n]]
```
Out[18]=
```
FAIL
```

In[19]:=
```
GuessRE[dataP, a[n], {0, 3}, {0, 0}]
```

Out[19]=
```
{{-a[n] - a[1 + n] + a[3 + n] == 0, a[0] == 3, a[1] == 0, a[2] == 2}, ogf}
```