Advanced Basics of *Mathematica* Grammar 2011 (easy/medium)

Replacements

Very important equations arising in the study on quantum integrable models are the so called Y-system equations:

$Y_j(u+1)\,Y_j(u-1) = \left(1+Y_{j-1}(u)\right)\left(1+Y_{j+1}(u)\right)$ for $j = 1, \ldots, n$ and $Y_j(u) = 0$ for $j \le 0$ or $j \ge n+1$.

By considering a few different n's convince yourself that they imply that

$Y_j(u+n+3) = Y_{n-j+1}(u)$

Hint: Understand

```
f[5] /. f[a_] :→ f[a - 1] + a /; a > 0
```

```
f[5] //. f[a_] :→ f[a - 1] + a
```

```
f[5] //. f[a_] :→ f[a - 1] + a /; a > 0
```

Solution

Conjugate

Consider some generic complex function such as

```
s = (Exp[I kx ] + Exp[2 I ky ])
```

Sometimes we might want *Mathematica* to conjugate such expressions assuming all variables to be real unless told otherwise. That is we would like a function **conj** such that

```
conj[s] s // FullSimplify
```

should yield

2 (1 + Cos[kx - 2 ky])

Create such function. Hint: Run

```
FullForm /@ {Exp[I kx], A + B I, -2 I}
```

Solution

Tensor Products

How to proceed to get the following nice implementation of the tensor product:

```
(a b; c d) ⊗ (α β γ; ϵ ϕ ρ) // MatrixForm
```

$$\begin{pmatrix} a\alpha & a\beta & a\gamma & b\alpha & b\beta & b\gamma \\ a\epsilon & a\phi & a\rho & b\epsilon & b\phi & b\rho \\ c\alpha & c\beta & c\gamma & d\alpha & d\beta & d\gamma \\ c\epsilon & c\phi & c\rho & d\epsilon & d\phi & d\rho \end{pmatrix}$$

- Solution

Simplify

FullSimplify measures the complexity of the expression, which it tries to reduce then, in some very special way. It uses the function **LeafCount** . In some cases this leads to a really strange behavior:

Run

```
FullSimplify[Sin[3 (u + π)] + Cos[5 (u + π)]]
```

and

```
FullSimplify[Sin[3 (u + π)] - Cos[5 (u + π)]]
```

Apply **TreeForm** to both functions (with and without FullSimplify). Do it also for what you would like the output of the first line to be. Same for **LeafCount**. *Mathematica* wants to minimize **LeafCount**. Why doesn't *Mathematica* simplify the first line *as we would?*

Try to find some other examples where FullSimplify "fails".

The LeadCount function can be replaced by a user defined function. You can invent your own measure of complexity which you like. Hive several examples of your own **ComplexityCount** function and use it to simplify the expression you found.

Create your own function **Simpy** which simplifies expressions with 3π etc inside trigonometric functions in a more human way.

Hint: Functions which you might find useful are: TrigToExp, ExpandAll, ...

- Solution

Sqrt killer

Define a function which will bring the expression with a single square roots in denominator to the canonical form, that is

$$\frac{\ldots + \sqrt{\ldots}}{\left(\ldots + \sqrt{\ldots}\right) \ldots \left(\ldots + \sqrt{\ldots}\right)} \rightarrow \ldots + \ldots \sqrt{\ldots}$$

- Solution

Memorizer

Define a function which will solve

$$F_n == F_{n-1} + 1 / F_{n-2} \quad \text{with} \quad F_1 = F_2 = 1.2$$

Plot the sequence $\log(F_n)$ with $n = 1, \ldots, 1000$ (should take less than one second)

Solution

Puzzles*

***(from http : // richardwiseman.wordpress.com/ , solutions proposed here taken from the comment box and were given by Simon)**

Puzzle 1: How can you place the arithmetical signs '+' and '-' between the consecutive numbers 123456789 so that the end result is 100?
Try to come up with a code to solve this problem. Also, decode the sollowing solution*

```
Do[If[ToExpression[str = StringJoin[Riffle[Characters["123456789"], i]]] == 100,
  Print["100=", str]], {i, Tuples[{"+", "-", ""}, 8]}]
```

Puzzle 2: Yesterday I went shopping and picked up four items. When I got to the till, the cashier added the price of the four items together and the bill was £7.11. I then noticed that I would get exactly the same total if I were to multiply the four prices. How much did each of the items cost?
Note: What the puzzle should say is: "Find integers a,b,c,d such that a+b+c+d = 711 and a b c d = 711 000 000."
Again, try to come up with a code to solve this problem and/or decode the solution*

```
Select[IntegerPartitions[711, {4}], Times @@ # == 711 000 000 &]
```

Solution given in the text