QuantumGroups`Algebra` package

A subpackage for QuantumGroups v2.
Version 2.0, June 20, 2005, Scott Morrison

Introduction

Implementation

```
BeginPackage["QuantumGroups`Algebra`",
 {"QuantumGroups`", "QuantumGroups`RootSystems`", "QuantumGroups`WeylGroups`"}]
```

```
X; K; 1; 0;
```

```
SuperPlus; SuperMinus;
```

```
PositiveGenerators;
NegativeGenerators;
CartanGenerators;
Generators;
NonCommutativePower;
Δ;
Δop;
OperatorWeight;
OperatorLength;
```

```
Begin["`Private`"];
```

```
PositiveGenerators[Γ_] := Table[SuperPlus[Xi], {i, 1, Rank[Γ]}];
```

```
NegativeGenerators[Γ_] := Table[SuperMinus[Xi], {i, 1, Rank[Γ]}];
```

```
CartanGenerators[Γ_] := Table[Ki, {i, 1, Rank[Γ]}];
```

```
Generators[Γ_] :=
 CartanGenerators[Γ] ~ Join ~ PositiveGenerators[Γ] ~ Join ~ NegativeGenerators[Γ]
```

Composition

```
Unprotect[NonCommutativeMultiply];
```

```
0 ** b_ := 0
```

```
a_ ** 0 := 0
```

```
a_ ** b_Plus := a ** # & /@ b
```

```
a_Plus ** b_ := # ** b & /@ a
```

```
(α_?qNumberQ a_) ** b_ := α (a ** b)
a_ ** (α_?qNumberQ b_) := α (a ** b)
```

```
1 ** Z_ := Z
Z_ ** 1 := Z
```

```
(a_ ⊗ b_) ** (c_ ⊗ d_) := (a ** c) ⊗ (b ** d)
```

```
NonCommutativePower[x_, n_?(# ≥ 2 &)] := NonCommutativeMultiply @@ Table[x, {n}]
```

```
NonCommutativePower[x_, 1] := x
```

```
NonCommutativePower[x_, 0] := 1
```

```
NonCommutativePower[x_, n_?(# < 0 &)] := NonCommutativePower[x^-1, -n]
```

```
Protect[NonCommutativeMultiply];
```

Comultiplication

```
Δ[1] = 1 ⊗ 1;
Δ[K_i_] := K_i ⊗ K_i
Δ[K_i_^-1] := K_i^-1 ⊗ K_i^-1
Δ[SuperPlus[X_i_]] := SuperPlus[X_i] ⊗ K_i + 1 ⊗ SuperPlus[X_i]
Δ[SuperMinus[X_i_]] := SuperMinus[X_i] ⊗ 1 + K_i^-1 ⊗ SuperMinus[X_i]
```

```
Δ[α_?qNumberQ A_] := α Δ[A]
```

```
Δ[A_ ** B_] := Δ[A] ** Δ[B]
```

```
Δop[Z_] := Δ[Z] /. X_ ⊗ Y_ :> Y ⊗ X
```

```
Δ[X_Plus] := Δ /@ X
Δ[X_CircleTimes] := Δ[First[X]] ⊗ Rest[X]
```

Weights of quantum group generators

```
OperatorWeight[Γ_][0] := ZeroVector[Rank[Γ]]
OperatorWeight[Γ_][1] := ZeroVector[Rank[Γ]]
OperatorWeight[Γ_][K_i_] := ZeroVector[Rank[Γ]]
OperatorWeight[Γ_][K_i_^-1] := ZeroVector[Rank[Γ]]
```

```
OperatorWeight[Γ_][SuperPlus[X_i_]] :=
 OperatorWeight[Γ][SuperPlus[X_i]] = SimpleRoots[Γ][[i]]
OperatorWeight[Γ_][SuperMinus[X_i_]] :=
 OperatorWeight[Γ][SuperMinus[X_i]] = -SimpleRoots[Γ][[i]]
```

```
OperatorWeight[Γ_][A_ ** B_] := OperatorWeight[Γ][A] + OperatorWeight[Γ][B]
```

```
OperatorWeight[Γ_][A_ ⊗ B_] := OperatorWeight[Γ][A] + OperatorWeight[Γ][B]
```

```
OperatorWeight[Γ_][α_?qNumberQ A_] := OperatorWeight[Γ][A]
(*OperatorWeight[Γ_][A_+B_]/;(OperatorWeight[Γ][A]==OperatorWeight[Γ][B]):=
 OperatorWeight[Γ][A]*)
OperatorWeight[Γ_][A_Plus] /; (Length[Union[OperatorWeight[Γ] /@ (List @@ A)]] == 1) :=
 OperatorWeight[Γ][First[A]]
```

```
OperatorLength[Γ_, A_] :=
 OperatorLength[Γ, A] = With[{λ = OperatorWeight[Γ][A]}, √KillingForm[Γ][λ, λ]]
```

K_ρ

```
K_{A_1,ρ} = K_1;
K_{Γ_,ρ} := Inner[NonCommutativePower, CartanGenerators[Γ],
   (Plus @@ PositiveRoots[Γ]).Inverse[SimpleRoots[Γ]], NonCommutativeMultiply]
```

```
End[];
```

```
EndPackage[];
```