**Function:** Octonion:-associator - returns the associator value of three octonions,  
Octonion:-commutator - returns the commutator value of two octonions  
Octonion:-Phi - associative 3-form of three octonions

**Calling Sequence:**

`associator(p1, p2, p3);`  
`commutator(p1, p2);`  
`Phi(p1, p2, p2);`

**Parameters:**

`p1, p2, p3` - polynomials of type 'octonion'

**Description:**

- The associator of three octonions `p1`, `p2`, and `p3` is defined as:

  \[
  \text{associator}(p1, p2, p3) = (p1 \&o p2) &o p3 - p1 &o (p2 &o p3).
  \]

- The commutator of two octonions `p1` and `p2` is defined as:

  \[
  \text{commutator}(p1, p2) = p1 &o p2 - p2 &o p1.
  \]

- The associative 3-form `Phi` of three octonions is defined as:

  \[
  \Phi(p1, p2, p3) = \frac{1}{2} \text{realpart}(p1 &o (p2_{bar} &o p3) - p3 &o (p2_{bar} &o p1))
  \]

  where `p2_{bar} = o_conjug(p2)`.

- For information about type 'octonion' see `type/octonion`.

**Examples:**

```maple
> restart: with(Clifford): with(Octonion);

Warning, the protected name version has been redefined and unprotected

[\Phi, \text{associator}, \text{commutator}, \text{def_omultable}, \text{o_conjug}, \text{oinv}, \text{omul}, \text{omultable}, \text{onorm},
  \text{oversion}, \text{purevectorpart}, \text{realpart}]

> p1 := 1 - 2*e1 + e4 + 3*e6 - e7; p2 := 2 - e1 + e3 + 2*e6 - e7; p3 := 2*e2 + e3 + 3*e5 - e6;

  p1 := 1 - 2\ e1 + e4 + 3\ e6 - e7
  p2 := 2 - e1 + e3 + 2\ e6 - e7
  p3 := 2\ e2 + e3 + 3\ e5 - e6

> type(p1, octonion); type(p2, octonion); type(p3, octonion);```

Octonion multiplication is not associative:

```plaintext
> associator(p1,p2,p3);
20 e3 + 14 e4 – 6 e5 – 2 e6 + 24 e7 – 8 e1 – 2 e2
```

However, when p1, p2, and p3 are considered as elements in the Clifford algebra Cl(0,7), which is associative, we get:

```plaintext
> (p1 &c p2) &c p3 - p1 &c (p2 &c p3);
0
> commutator(p1,p2);
2 e3 + 6 e4 + 4 e5 – 2 e6 – 4 e7 + 2 e1 – 4 e2
> Phi(p1,p2,p3);
4
```

See Also: Clifford:-`&c`, def_omultable, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
Function: Octonion:-associator - returns the associator value of three octonions,
Octonion:-commutator - returns the commutator value of two octonions
Octonion:-Phi - associative 3-form of three octonions

Calling Sequence:
associator(p1,p2,p3);
commutator(p1,p2);
Phi(p1,p2,p3);

Parameters:
p1, p2, p3 - polynomials of type 'octonion'

Description:
• The associator of three octonions p1, p2, and p3 is defined as:

\[ \text{associator}(p1,p2,p3) = (p1 \&o p2) \&o p3 - p1 \&o (p2 \&o p3). \]

• The commutator of two octonions p1 and p2 is defined as:

\[ \text{commutator}(p1,p2) = p1 \&o p2 - p2 \&o p1. \]

• The associative 3-form Phi of three octonions is defined as:

\[ \Phi(p1,p2,p3) = \frac{1}{2} \text{realpart}(p1 \&o (p2\_bar \&o p3) - p3 \&o (p2\_bar \&o p1)) \]

where \( p2\_bar = o\_conjug(p2) \).

• For information about type 'octonion' see `type/octonion`.

Examples:
```maple
restart:with(Clifford):with(Octonion);
Warning, the protected name version has been redefined and unprotected

[Φ, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm,
 overseversion, purevectorpart, realpart]

> p1:=-1-2*e1+e4+3*e6-e7; p2:=-2-e1+e3+2*e6-e7; p3:=-2*e2+e3+3*e5-e6;
  p1 := 1 - 2 e1 + e4 + 3 e6 - e7
  p2 := 2 - e1 + e3 + 2 e6 - e7
  p3 := 2 e2 + e3 + 3 e5 - e6

> type(p1,octonion); type(p2,octonion); type(p3,octonion);
```
Octonion multiplication is not associative:

\[ \text{associator}(p_1, p_2, p_3); \]
\[-8e_1 - 2e_2 + 20e_3 + 14e_4 - 6e_5 - 2e_6 + 24e_7 \]

However, when \( p_1, p_2, \) and \( p_3 \) are considered as elements in the Clifford algebra \( \text{Cl}(0,7) \), which is associative, we get:

\[ (p_1 &c p_2) &c p_3 - p_1 &c (p_2 &c p_3); \]
\[ 0 \]

\[ \text{commutator}(p_1, p_2); \]
\[ 2e_1 - 4e_2 + 2e_3 + 6e_4 + 4e_5 - 2e_6 - 4e_7 \]

\[ \Phi(p_1, p_2, p_3); \]
\[ 4 \]

See Also: Clifford:-&c, def_omultable, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
Function: Octonion:-def_omultable - define octonionic multiplication table

Calling Sequence:
def_omultable(F);

Parameters:
F - a list of type `Fano_triples`

Description:

• Procedure 'def_omultable' allows user to define an octonionic multiplication table which could be
different than the default one.

• The default multiplication table is initialized at the time when the 'OCTONION' package is being
loaded. It can also be re-defined by issuing the following command:

  > def_omultable(_default_Fano_triples);

where _default_Fano_triples is a global list with default Fano triples. See `type/Fano_triples` for
more information.

• Use `omultable` to display currently defined multiplication table.

• Use Clifford:-CLIFFORD_ENV to display current environmental variables used by 'CLIFFORD'
and 'Octonion'.

Examples:

  > restart:with(Clifford):with(Octonion);

  Warning, the protected name version has been redefined and unprotected

  [Φ, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm,
   oversion, purevectorpart, realpart]

  > omultable(); #default multiplication table

  \[
  \begin{bmatrix}
  -1d & e4 & e7 & -e2 & e6 & -e5 & -e3 \\
  -e4 & -1d & e5 & e1 & -e3 & e7 & -e6 \\
  -e7 & -e5 & -1d & e6 & e2 & -e4 & e1 \\
  e2 & -e1 & -e6 & -1d & e7 & e3 & -e5 \\
  -e6 & e3 & -e2 & -e7 & -1d & e1 & e4 \\
  e5 & -e7 & e4 & -e3 & -e1 & -1d & e2 \\
  e3 & e6 & -e1 & e5 & -e4 & -e2 & -1d \\
  \end{bmatrix}
  \]

  For example, we get the first row as follows:

  > seq(e1 &o e||i,i=1..7);

  [Id, e4, e7, -e2, e6, -e5, -e3]

  The second row we get as follows:

  > seq(e2 &o e||i,i=1..7);
and so on.

Multiplication table can be erased as follows:

```latex
code
> subsop(4=NULL, eval(omul)):
> omultable();
Ocnonion multiplication table is not currently defined. Use 'def_omultable' to
define a new table.
```

Finally, we re-initialize the table using the default Fano triples:

```latex
code
> _default_Fano_triples;
```

```
], [1, 3, 7], [1, 2, 4], [1, 5, 6], [2, 3, 5], [2, 6, 7], [3, 4, 6], [4, 5, 7])
> def_omultable(_default_Fano_triples);
> omultable();
```

```
\begin{tabular}{cccccc}
\hline
\(-e_4, -e_5, e_1, -e_3, e_7, -e_6\
\hline
\(-e_4, -e_5, e_1, -e_3, e_7, -e_6\)
\(-e_7, -e_5, -e_6, e_2, -e_4, e_1\)
\(-e_2, -e_1, -e_6, -e_7, e_3, -e_5\)
\(-e_6, e_3, -e_2, -e_7, -e_1, e_4\)
\(-e_5, -e_7, e_4, -e_3, -e_1, -e_6, e_2\)
\(-e_3, e_6, -e_1, e_5, -e_4, -e_2, -e_1, -e_6\)
\hline
\end{tabular}
```

However, the following is another valid list of Fano triples:

```latex
code
> new_Fano_triples:=[[6,2,5], [6,3,4], [6,7,1], [2,3,7], [3,1,5], [2,4 ,1], [4,5,7]];
> type(new_Fano_triples,Fano_triples);
true
> def_omultable(new_Fano_triples);
> omultable();
```

```
\begin{tabular}{cccccc}
\hline
\(-e_4, -e_5, e_1, -e_3, e_7, -e_6\)
\(-e_7, -e_5, -e_6, e_2, -e_4, e_1\)
\(-e_2, -e_1, -e_6, -e_7, e_3, -e_5\)
\(-e_6, e_3, -e_2, -e_7, -e_1, e_4\)
\(-e_5, -e_7, e_4, -e_3, -e_1, -e_6, e_2\)
\(-e_3, e_6, -e_1, e_5, -e_4, -e_2, -e_1, -e_6\)
\hline
\end{tabular}
```

which is a different multiplication table than before.

- See Also: `type/Fano_triples`, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
**Function:** Octonion:-`type/Fano_triples` - a list of lists used to define octonionic multiplication table

**Calling Sequence:**

```
type(F,Fano_triples);
```

**Parameters:**

`F` - a list of lists

**Description:**

- A list of lists `F` is of type 'Fano_triples' if:
  1. the list `F` contains seven lists `F1, F2, F3, F4, F5, F6, and F7`;
  2. each of the seven lists `F1, ... , F7` contains three integers from the set `{1,2,3,4,5,6,7}`;
  3. each of the seven integers `{1,2,3,4,5,6,7}` appears in exactly three of the seven lists `F1, ... , F7`.

- A default list of Fano triples is stored in a global list `_default_Fano_triples`.

- A valid list of seven Fano triples may be used to label seven points and seven lines in the Fano plane `F_2`.

- The set of integers `{1,2,3,4,5,6,7}` is used because we use `{e1,e2,e3,e4,e5,e6,e7}` for the pure octonion basis.

- If `[i,j,k]` is one of the seven valid Fano triples `F1, ... , F7`, then:
  1. `omul(ei,ej) = ek`, `omul(ej,ek) = ei`, `omul(ek,ei) = ej`;
  2. `omul(ej,ei) = -ek`, `omul(ek,ej) = -ei`, `omul(ei,ek) = -ej`;

- The default multiplication table is initialized at the time when the 'Octonion' package is being loaded. It can also be re-defined by issuing the following command:

  ```
  > def_omultable(_default_Fano_triples);
  ```

  where `_default_Fano_triples` is a global list with default Fano triples. See `type/Fano_triples` for more information.

- Use `omultable` to display currently defined multiplication table.

- See `omul` for octonionic multiplication.

- To display all environmental variables used by 'CLIFFORD' and 'Octonion' packages, use `Clifford:-CLIFFORD_ENV`.

**Examples:**

```
> restart:with(Clifford):with(Octonion);

Warning, the protected name version has been redefined and unprotected

[Φ, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm,
```
version, purevectorpart, realpart]
>
> _default_Fano_triples;#default Fano triples

[[1, 3, 7], [1, 2, 4], [1, 5, 6], [2, 3, 5], [2, 6, 7], [3, 4, 6], [4, 5, 7]]

For example, the first list implies the following about \{e_1, e_3, e_7\}:

> 'omul(e_1, e_3)' = omul(e_1, e_3); 'omul(e_3, e_7)' = omul(e_3, e_7); 'omul(e_7, e_1)'

= omul(e_7, e_1);

\[\begin{align*}
\text{omul}(e_1, e_3) &= e_7 \\
\text{omul}(e_3, e_7) &= e_1 \\
\text{omul}(e_7, e_1) &= e_3
\end{align*}\]

and

> 'omul(e_3, e_1)' = omul(e_3, e_1); 'omul(e_7, e_3)' = omul(e_7, e_3); 'omul(e_1, e_7)'

= omul(e_1, e_7);

\[\begin{align*}
\text{omul}(e_3, e_1) &= -e_7 \\
\text{omul}(e_7, e_3) &= -e_1 \\
\text{omul}(e_1, e_7) &= -e_3
\end{align*}\]

and so on.

However, the following is another valid list of Fano triples:

> new_Fano_triples := [[6, 2, 5], [6, 3, 4], [6, 7, 1], [2, 3, 7], [3, 1, 5], [2, 4, 1], [4, 5, 7]]

> type(new_Fano_triples, Fano_triples);

true

while the following is not:

> another_Fano_triples := [[4, 2, 5], [6, 3, 4], [6, 7, 1], [2, 3, 7], [3, 1, 5], [2, 4, 1], [4, 5, 7]]

> type(another_Fano_triples, Fano_triples);

false

The reason is that '4' appears in four lists.

> See Also: def_omultable, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.

Last revised: December 1, 2003 /RA
Function: Octonion:-o_conjug - octonionic conjugation in the octonionic algebra

Calling Sequence:

o_conjug(o);

Parameters:

o  - expression of the type 'octonion'

Description:

• Procedure 'o_conjug' computes octonionic conjugation in the octonionic algebra:

\[ o_{\text{conjug}}(x_0 + x) = x_0 - x \]

• where \( x_0 \) is a real number and \( x = x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \).

• Conjugation is an anti-automorphism of the octonionic algebra. This means that \( o_{\text{conjug}}(o_1 \& o_2) = o_{\text{conjug}}(o_2) \& o_{\text{conjug}}(o_1) \).

• For information about type 'octonion' see `type/octonion`.

Examples:

```maple
> restart; with(Clifford); with(Octonion);

> o1 := x0 + add(x||i*e||i, i=1..7);

> o2 := y0 + add(y||i*e||i, i=1..7);

> L := o_conjug(omul(o1, o2));

> R := omul(o_conjug(o2), o_conjug(o1));

> simplify(L - R);

0

> o1inv := o_conjug(o1);

> o1 &o o1inv;

> realpart(%);
```

Any octonion \( o1 \) times its conjugate is a scalar:
Since octonions are treated as paravectors in the Clifford algebra Cl(0,7), octonionic conjugate of any octonion can be obtained also by taking grade involution:

\[ \text{gradeinv}(o1); \]

\[ x0 \text{Id} - x1 e1 - x2 e2 - x3 e3 - x4 e4 - x5 e5 - x6 e6 - x7 e7 \]

However, grade involution in Cl(0,7) is not an antiautomorphism of Cl(0,7): it is an automorphism of Cl(0,7). Note: in the above output, the unit element in Cl(0,7) is denoted as 'Id'.

See Also: omul, oinv, Clifford:-q_conjug, Clifford:-conjugation, Clifford:-gradeinv, realpart

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved. Last revised: December 1, 2003 /RA
Function: Octonion:-`type/octonion` - type octonion

Calling Sequence:

\texttt{type(p,octonion);}

Parameters:

- \texttt{p} - an expression of type 'algebraic'

Description:

- Any polynomial \( p \) expressible as follows

\[ p = x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \]

or

\[ p = x_0 \text{Id} + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \]

is of type 'octonion'.

- The unit element in the octonion algebra may be entered as 1 or as 'Id'. In some computations 'Id' will be returned.

- Use \texttt{omultable} to display currently defined multiplication table.

- See \texttt{omul} for octonionic multiplication.

Examples:

\begin{verbatim}
> restart:with(Clifford):with(Octonion);

Phi, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm, oversion, purevectorpart, realpart

> p1:=1-2*e2+e4+e5+4*e6-e7;

\mathbf{p1} := 1 - 2 e_2 + e_4 + e_5 + 4 e_6 - e_7

> type(p1,octonion);

true

> p2:=p1+e8;

\mathbf{p2} := 1 - 2 e_2 + 4 e_4 + 4 e_5 + 6 e_6 - e_7 + e_8

> type(p2,octonion);

false
\end{verbatim}

See Also: def_omultable, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.

Last revised: December 1, 2003 /RA
Function: Octonion:-oinv - symbolic inverse in the octonionic division ring

Calling Sequence:
oinv(o);

Parameters:
o  - expression of the type 'octonion'

Description:
• Procedure 'oinv' calculates a symbolic inverse of any non-zero octonion. Recall that octonions form a non-associative, non-commutative division ring.
• For information about type 'octonion' see `type/octonion`.
• Note that any of the following is an illegal entry: 1/e1, e1\(^{-1}\), etc.
• Recall that octonionic product can be computed with the procedure omul.

Examples:
> restart:with(Clifford):with(Octonion);
Warning, the protected name version has been redefined and unprotected

Φ
associator commutator def_omultable o_conjug oinv omul omultable onorm
oversion purevectorpart realpart

> o1:=1-2*e1+3*e3+e4-e6+e7;
o1 := 1 - 2 e1 + 3 e3 + e4 - e6 + e7

> p:=e1+e2;pinv:=oinv(p);
p := e1 + e2
pinv := -e1/2 - e2/2

> omul(p,pinv);
Id

> o1inv:=oinv(o1); #inverse of o1

> omul(o1inv,o1); #checking that o1inv is the inverse of o1
Id

> o2:=x0+add(x||i*e||i,i=1..7);
o2 := x0 + x1 e1 + x2 e2 + x3 e3 + x4 e4 + x5 e5 + x6 e6 + x7 e7

> o2inv:=oinv(o2); #symbolic inverse of o2
\[ o2inv := \frac{x0}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x1 e1}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x2 e2}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x3 e3}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x4 e4}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x5 e5}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x6 e6}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} - \frac{x7 e7}{x0^2 + x1^2 + x5^2 + x6^2 + x7^2 + x2^2 + x3^2 + x4^2} \]

> omul(o2,o2inv);

\[ \text{Id} \]

See Also: onorm, omul, def_omultable, omultable

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
Function: Octonion:-omul - octonion product in the octonion non-associative division ring and its infix form '&o'

Calling Sequence:

omul(o1,o2,...on);
o1 &o o2 &o ... &o on;

Parameters:
o1, o2, ..., on - expressions of the type 'octonion'

Description:

• Procedure 'omul' and its infix form '&o' give the octonion product in the non-associative division ring of octonions.

• Octonions are considered here as para-vectors in the Clifford algebra Cl(0,7), that is, any expression of the form

x0 + x1*e1 + x2*e2 + x3*e3 + x4*e4 + x5*e5 + x6*e6 + x7*e7

where x0, x1, ..., x7 are real numbers, is of type 'octonion'. See `type/octonion` for more information.

• The basis elements for the octonion algebra are \{1,e1,e2,e3,e4,e5,e6,e7\} (sometimes 'Id' is returned instead of '1'). They are collected in a global variable '_octbasis'. The basis elements \{e1,e2,e3,e4,e5,e6,e7\} give pure octonions and are collected in a global variable _pureoctbasis.

• To display environmental variables from CLIFFORD and Octonion, use Clifford:-CLIFFORD_ENV.

• The infix form is given by ')&o', e.g., omul(e1,e2) = e1 &o e2. Remember that 'omul' is non-associative!

• Octonionic inverse is computed with oinv.

• To speed up computations, set the global variable _prolevel to 'true'. To find out more, see help page on Clifford:-cliparse.

• To see the default multiplication table try omultable and to define your own octonionic multiplication see def_omultable.

Examples:

> restart:with(Clifford):with(Octonion);

Warning, the protected name version has been redefined and unprotected

[Φ, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm,
  oversion, purevectorpart, realpart]

The following is the default octonionic multiplication table:
> omultable();

\[
\begin{bmatrix}
-Id & e4 & e7 & -e2 & e6 & -e5 & -e3 \\
-e4 & -Id & e5 & e1 & -e3 & e7 & -e6 \\
-e7 & -e5 & -Id & e6 & e2 & -e4 & e1 \\
e2 & -e1 & -e6 & -Id & e7 & e3 & -e5 \\
e6 & e3 & -e2 & -e7 & -Id & e1 & e4 \\
e5 & -e7 & e4 & -e3 & -e1 & -Id & e2 \\
e3 & e6 & -e1 & e5 & -e4 & -e2 & -Id
\end{bmatrix}
\]

> o1:=1-2*e1+3*e3+e4-e6+e7;

\[
o1 := 1 - 2 e1 + 3 e3 + e4 - e6 + e7
\]

> o2:=2+e3-4*e6+e7;

\[
o2 := 2 + e3 - 4 e6 + e7
\]

> type(o1,octonion),type(o2,octonion);

Cliplus has been loaded. Definitions for type/climon and type/clipolynom now in clude &C and &C[K]. Type ?cliprod for help.

\[
true, true
\]

> omul(o1,o2);

\[
-2 e1 - 6 Id + 5 e3 + 13 e4 - 7 e6 + e7 - 9 e5 + 3 e2
\]

Octonionic multiplication is not commutative:

> o1 &o o2;

\[
-2 e1 - 6 Id + 5 e3 + 13 e4 - 7 e6 + e7 - 9 e5 + 3 e2
\]

> o2 &o o1;

\[
-6 e1 + 5 e7 + 9 e5 + 9 e3 - 6 Id - 5 e6 - 9 e4 - 3 e2
\]

We show now that it is not associative either:

> (e1 &o e2) &o e3;

\[
-e6
\]

> e1 &o (e2 &o e3);

\[
e6
\]

> o3:=2-3*e1+e5-e7;

\[
o3 := 2 - 3 e1 + e5 - e7
\]

> (o1 &o o2) &o o3;

\[
16 e1 - 8 Id - 21 e2 + 2 e3 + 43 e4 + 10 e5 - 40 e6 + 36 e7
\]

> o1 &o (o2 &o o3);

\[
6 e1 - 8 Id + 47 e2 + 8 e3 + 5 e4 - 18 e5 - 38 e6 + 38 e7
\]

The difference between (o1 &o o2) &o o3 and o1 &o (o2 &o o3) is measured by an associator, or see associator:

> associator(o1,o2,o3);

\[
10 e1 - 68 e2 - 6 e3 + 38 e4 + 28 e5 - 2 e6 - 2 e7
\]

The difference between o1 &o o2 and o2 &o o1 is measured by a commutator, or see commutator:

> commutator(o1,o2);

\[
4 e1 - 4 e3 + 22 e4 - 2 e6 - 4 e7 - 18 e5 + 6 e2
\]
See Also: Clifford:-version, oinv, def_omultable, omultable

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
Function: Octonion:-omultable - display current octonionic multiplication table

Calling Sequence:
omultable();

Parameters:
no parameters needed

Description:
• Procedure 'omultable' displays current octonionic multiplication table or returns a message informing that the table has not been defined.

• When the Octonion package is loaded, the default multiplication table is initialized. This default table is defined by a default list of Fano triples (see `type/Fano_triples`) which are stored in a global variable _default_Fano_triples.

• To see environmental variables used in 'CLIFFORD' and 'Octonion', see procedure Clifford:-CLIFFORD_ENV.

• The multiplication table is displayed in a form of a 7 by 7 matrix such that its (i,j)-entry, i,j=1,...,7, gives the octonion product of ei and ej, that is, the product ei &o ej.

• Recall that the elements of the pure octonion basis {e1,e2,e3,e4,e5,e6,e7} are stored in a global list _pureoctbasis.

• To speed up computations, procedure 'omul', which gives the octonionic product (see omul) has a remember table. This remember table can be erased using the command subsop(4=NULL,eval(omul)).

• Octonionic multiplication can be re-defined by the user using the procedure def_omultable.

Examples:
> restart:with(Clifford):with(Octonion);
Warning, the protected name version has been redefined and unprotected

> oversion();

'Octonion' - A Maple 9 Package for Computations with Octonions (version 9)
Last revised: December 1, 2003
Copyright 1995-2004, by Rafal Ablamowicz, Tennessee Technological University
Department of Mathematics, Box 5054
Tennessee Technological University, Cookeville, TN 38505
> omultable();

\[
\begin{bmatrix}
    -Id & e4 & e7 & -e2 & e6 & -e5 & -e3 \\
    -e4 & -Id & e5 & e1 & -e3 & e7 & -e6 \\
    -e7 & -e5 & -Id & e6 & e2 & -e4 & e1 \\
    e2 & -e1 & -e6 & -Id & e7 & e3 & -e5 \\
    -e6 & e3 & -e2 & -e7 & -Id & e1 & e4 \\
    e5 & -e7 & e4 & -e3 & -e1 & -Id & e2 \\
    e3 & e6 & -e1 & e5 & -e4 & -e2 & -Id
\end{bmatrix}
\]

For example, we get the first row as follows:

> seq(e1 &o e||i,i=1..7);

\[-Id, e4, e7, -e2, e6, -e5, -e3\]

The second row we get as follows:

> seq(e2 &o e||i,i=1..7);

\[-e4, -Id, e5, e1, -e3, e7, -e6\]

and so on.

Multiplication table can be erased as follows:

> subsop(4=NULL, eval(omul)):
> omultable();

Octonion multiplication table is not currently defined. Use 'def_omultable' to define a new table.

When the multiplication table has been erased, computations still can be performed using the approach described in Pertti Lounesto's 'Clical'. In fact, the default multiplication table is the one used by Lounesto's. However, they will take longer to accomplish. For example:

> seq(e1 &o e||i,i=1..7);

Cliplus has been loaded. Definitions for type/climon and type/clipolynom now in clude &C and &C[K]. Type ?cliprod for help.

\[-Id, e4, e7, -e2, e6, -e5, -e3\]

which yields the same result as before.

Finally, we re-initialize the table using the default Fano triples:

> _default_Fano_triples;

\[[1, 3, 7], [1, 2, 4], [1, 5, 6], [2, 3, 5], [2, 6, 7], [3, 4, 6], [4, 5, 7]\]

> def_omultable(_default_Fano_triples);
> omultable();
Thus, the table has been re-initialized.

See Also: `type/Fano_triples`, `def_omultable`, `omul`

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
Function: Octonion:-onorm - norm of an octonion

Calling Sequence:

onorm(o);

Parameters:

o - expression of the type 'octonion'

Description:

- Procedure 'onorm' calculates norm of an octonion o. It is defined as follows:

  \[ \text{onorm}(o) = \sqrt{o \& o_\text{conjug}(o)} = \sqrt{x_0^2 + x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2} \]

  where \( o = x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \), and \( x_0, x_1, ..., x_7 \), are real parameters.

- Recall that octonionic product can be computed with the procedure \texttt{omul} or with its infix form '\&o'.

- For information about type 'octonion' see \texttt{`type/octonion'}.  

Examples:

\[
\begin{align*}
> \text{restart;} \text{with(Clifford);} \text{with(Octonion);} \\
& \text{Warning, the protected name version has been redefined and unprotected} \\
& \Phi, \text{associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm,} \\
& \text{oversion, purevectorpart, realpart} \\
& > o1 := 1 - 2 e_1 + 3 e_3 + e_4 - e_6 + e_7; \\
& \quad o1 := 1 - 2 e_1 + 3 e_3 + e_4 - e_6 + e_7 \\
& > \text{onorm(o1);} \quad \# \text{norm of o1} \\
& \text{Cliplus has been loaded. Definitions for type/climon and type/clipolynom now in clude &C and &C[K]. Type ?cliprod for help.} \\
& \quad \sqrt{17} \\
& > o2 := 2 - 3 e_4 + e_5 + 4 e_6 - e_7; \\
& \quad o2 := 2 - 3 e_4 + e_5 + 4 e_6 - e_7 \\
& > \text{onorm(o2);} \\
& \quad \sqrt{31} \\
\end{align*}
\]

Theorem [The Eight-Square Identity]

The norm in the octonion algebra is a ring homomorphism.

\[
\begin{align*}
& > o1 := x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7; \\
& \quad o1 := x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \\
\end{align*}
\]
> o2:=y0+y1*e1+y2*e2+y3*e3+y4*e4+y5*e5+y6*e6+y7*e7;

\[ o2 := y0 + y1 \, e1 + y2 \, e2 + y3 \, e3 + y4 \, e4 + y5 \, e5 + y6 \, e6 + y7 \, e7 \]

We will now verify that

\[ \text{onorm}(o1 \, \& \, o2) = \text{onorm}(o1) \, \, \text{onorm}(o2). \]

> factor(onorm(o1 & o2));

\[ \sqrt{(y^2 + y_6^2 + y_0^2 + y_5^2 + y_3^2 + y_2^2 + y_1^2 + y_7^2) \,(x^2 + x_5^2 + x_6^2 + x_2^2 + x_7^2 + x_1^2 + x_0^2 + x_3^2)} \]

> onorm(o1)*onorm(o2);

\[ \sqrt{x^2 + x_5^2 + x_6^2 + x_2^2 + x_7^2 + x_1^2 + x_0^2 + x_3^2} \, \sqrt{y^2 + y_6^2 + y_0^2 + y_5^2 + y_3^2 + y_2^2 + y_1^2 + y_7^2} \]

See Also: `oversion`, `omul`, `oinv`, `def_omultable`, `omultable`
Function: Octonion:-oversion - display information about the current version of the 'Octonion' package

Calling Sequence:

oversion();

Parameters:

no parameters needed

Description:

• Procedure 'oversion' displays information about the current version of the 'Octonion' package.

• The 'Octonion' package must be loaded after the 'CLIFFORD' package has been loaded. Therefore, in order to avoid confusion with the procedure Clifford:-version, this procedure is called 'oversion'.

• To display 'CLIFFORD' and 'Octonion' environmental variables, use procedure Clifford:-CLIFFORD_ENV.

• To multiply octonionic matrices, see Clifford:-rmulm.

Examples:

> restart:with(Clifford):with(Octonion);
Warning, the protected name version has been redefined and unprotected

Φ associator commutator def_omultable o_conjug oinv omul omultable onorm
purevectorpart realpart

> version(); #current version of CLIFFORD

CLIFFORD - A Maple 9 Package for Clifford Algebras
(Version 9 with global variable _prolevel and "Bigebra" package)
"Bigebra" package written with Bertfried Fauser, Universit"at Konstanz
Last revised: December 1, 2003 (Source file: clifford_M9_04.mws)
Copyright 1995-2004 by Rafal Ablamowicz (*) and Bertfried Fauser ($)"
If you are a Clifford algebra pro, assign 'true' to `_prolevel` and see how much faster your computations will be! But watch your syntax!

Use 'useproduct' to change value of _default_Clifford_product in Cl(B) from cmulRS when B is symbolic to cmulNUM when B is numeric. Type ?cmul for help.

> oversion(); #current version of Octonion

This is CLIFFORD version 9, library file : Clifford.m

> oversion(); #current version of Octonion

'Octonion' - A Maple 9 Package for Computations with Octonions (version 9)

Last revised: December 1, 2003

Copyright 1995-2004, by Rafal Ablamowicz, Tennessee Technological University

Department of Mathematics, Box 5054
Tennessee Technological University, Cookeville, TN 38505
phone: USA (931) 372-3569, fax: USA (931) 372-6353
e-mail: rablamotoicz@tnitech.edu
http://math.tntech.edu/rafal/clip9/

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.

See Also: omul

Last revised: December 1, 2003 /RA
**Function:** Octonion:-associator - returns the associator value of three octonions,
Octonion:-commutator - returns the commutator value of two octonions
Octonion:-Phi - associative 3-form of three octonions

**Calling Sequence:**

associator(p1,p2,p3);
commutator(p1,p2);
Phi(p1,p2,p3);

**Parameters:**
p1, p2, p3 - polynomials of type 'octonion'

**Description:**

- The associator of three octonions p1, p2, and p3 is defined as:

  \[
  \text{associator}(p1,p2,p3) = (p1 \&o p2) \&o p3 - p1 \&o (p2 \&o p3).
  \]

- The commutator of two octonions p1 and p2 is defined as:

  \[
  \text{commutator}(p1,p2) = p1 \&o p2 - p2 \&o p1.
  \]

- The associative 3-form Phi of three octonions is defined as:

  \[
  \Phi(p1,p2,p3) = \frac{1}{2} \text{realpart}(p1 \&o (p2_{\text{bar}} \&o p3) - p3 \&o (p2_{\text{bar}} \&o p1))
  \]

  where \( p2_{\text{bar}} = o_{\text{conjug}}(p2) \).

- For information about type 'octonion' see `type/octetonion`.

**Examples:**

```maple
> restart:with(Clifford):with(Octonion);
Warning, the protected name version has been redefined and unprotected

[Φ, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm, oversion, purevectorpart, rea
p1:=1-2*e1+e4+3*e6-e7; p2:=2-e1+e3+2*e6-e7; p3:=2*e2+e3+3*e5-e6;
  p1 := 1 - 2 e1 + e4 + 3 e6 - e7
  p2 := 2 - e1 + e3 + 2 e6 - e7
  p3 := 2 e2 + e3 + 3 e5 - e6
> type(p1/octetonion); type(p2/octetonion); type(p3/octetonion);
```
Cliplus has been loaded. Definitions for type/climon and type/clipolynom now in include &C and &C[K]. Type ?cliprod for help.

true
true
true

Octonion multiplication is not associative:

\[
\text{associator}(p1,p2,p3);
\]

\[ -8 e1 - 2 e2 + 20 e3 + 14 e4 - 6 e5 - 2 e6 + 24 e7 \]

However, when p1, p2, and p3 are considered as elements in the Clifford algebra Cl(0,7), which is associative, we get:

\[
\text{(p1 \&c p2) \&c p3 - p1 \&c (p2 \&c p3)};
\]

\[ 0 \]

\[
\text{commutator}(p1,p2);
\]

\[ 2 e1 - 4 e2 + 2 e3 + 6 e4 + 4 e5 - 2 e6 - 4 e7 \]

\[
\text{Phi}(p1,p2,p3);
\]

\[ 4 \]

See Also: Clifford:-`\&c`, def_omultable, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
**Function:** Octonion:-realpart - returns real part of any octonion,
Octonion:-purevectorpart - returns pure vector part of any octonion

**Calling Sequence:**

realpart(p);
purevectorpart(p);

**Parameters:**
p - a polynomial of type 'octonion'

**Description:**

- Any octonion p is expressible as

\[ p = x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \]

where \( x_0, x_1, \ldots, x_7 \) are real parameters.

- For information about type 'octonion' see `type/octonion`.

- The part \( x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \) of p is referred to as the 'pure vector part' of p.

- The coefficient \( x_0 \) is referred to as the 'real part' of p.

- Procedure 'realpart' is similar to Clifford:-scalarpart.

**Examples:**

```plaintext
> restart; 
with(Clifford): 
with(Octonion);

Warning, the protected name version has been redefined and unprotected

\[ \Phi, \text{associator, commutator, def\_omutable, o\_conjug, oinv, omul, omutable, onorm, oversion, purevectorpart, realpart} \]

\[ p1 := -2 e_1 + e_4 + 3 e_6 - e_7; \]

\[ p1 := 1 - 2 e_1 + e_4 + 3 e_6 - e_7 \]

\[ \text{type}(p1, \text{octonion}); \]
Cliplus has been loaded. Definitions for type/climon and type/clipolynom now in 
cluide &C and &C[K]. Type ?cliprod for help.

\[ \text{true} \]

\[ \text{realpart}(p1); \]
1

\[ \text{scalarpart}(p1); \]
1

\[ \text{purevectorpart}(p1); \]
\[-2 e_1 + e_4 + 3 e_6 - e_7\]
```
See Also: `def omultable`, `omultable`, `omul`

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
Function: Octonion:-realpart - returns real part of any octonion,
Octonion:-purevectorpart - returns pure vector part of any octonion

Calling Sequence:
realpart(p);
purevectorpart(p);

Parameters:
p - a polynomial of type 'octonion'

Description:
• Any octonion p is expressible as

\[ p = x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \]

or

\[ p = x_0 \text{Id} + x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \]

where \( x_0, x_1, \ldots, x_7 \), are real parameters.

• For information about type 'octonion' see `type/octonion`.

• The part \( x_1 e_1 + x_2 e_2 + x_3 e_3 + x_4 e_4 + x_5 e_5 + x_6 e_6 + x_7 e_7 \) of p is referred to as the 'pure vector part' of p.

• The coefficient \( x_0 \) is referred to as the 'real part' of p.

• Procedure 'realpart' is similar to Clifford:-scalarpart.

Examples:

```plaintext
restart:with(Clifford):with(Octonion);
Warning, the protected name version has been redefined and unprotected

[Φ, associator, commutator, def_omultable, o_conjug, oinv, omul, omultable, onorm,
  oversion, purevectorpart, realpart]

> p1:=1-2*e1+e4+3*e6-e7;

\[ p1 := 1 - 2 e_1 + e_4 + 3 e_6 - e_7 \]

> type(p1,octonion);

true

> realpart(p1);

1

> scalarpart(p1);

1

> purevectorpart(p1);

\[ -2 e_1 + e_4 + 3 e_6 - e_7 \]
```
See Also: def_omultable, omultable, omul

(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA
**Function:** Octonion:-setup - the initialization procedure for the package 'Octonion'

**Calling Sequence:**

none

**Parameters:**

none

**Description:**

- Procedure 'setup' is the initialization procedure for the 'Octonion' package. It is executed automatically when the package is loaded.
- At the time of loading, the following are defined:
  - `&o` - infix form for `omul`, the octonionic multiplication
  - `_octbasis = [Id, e1, e2, e3, e4, e5, e6, e7]` - standard octonion basis as Maple global variable in Cl(0,7)
  - `_pureoctbasis = [e1, e2, e3, e4, e5, e6, e7]` - pure octonion basis as Maple global variable in Cl(0,7)
  - `_default_Fano_triples = [[1,3,7],[1,2,4],[1,5,6],[2,3,5],[2,6,7],[3,4,6],[4,5,7]]` - default Fano triples that define octonionic multiplication
  - `_default_squares = [-Id, -Id, -Id, -Id, -Id, -Id, -Id]` - default squares of the pure octonionic basis

- To see all environmental variables that are defined and used by 'CLIFFORD', use procedure `Clifford:-CLIFFORD_ENV`.
- All procedures and types in 'Octonion' are protected.

**Examples:**

```maple
> restart:with(Clifford):with(Octonion):

Warning, the protected name version has been redefined and unprotected

> CLIFFORD_ENV();

`>>> Global variables defined in Clifford:-setup are now available and have the
se values: <<<``
`************* Start *************``
dim_V = 9
_default_Clifford_product = Clifford:-cmulNUM
_prolevel = false
Shortcut_in_minimalideal = true
Shortcut_in_Kfield = true
Shortcut_in_spinorKbasis = true
Shortcut_in_spinorKrepr = true
_warnings_flag = true
_scalartypes = {numeric, constant, function, rational, mathfunc, complex, indexed, ^, RootOf}
```
_quatbasis = [[Id, e3we2, e1we3, e2we1], {\`Maple has assigned qi:=\-e2we3, qj:=e1we3, qk:=\-e1we2\}`]

Cliplus has been loaded. Definitions for type/climon and type/clipolynom now in clude &C and &C[K]. Type ?cliprod for help.

```
\>> Global variables defined in Cliplus:-setup are now available and have thes e values: <<<
```
```
\-------------- Start  ---------------
macro(cmul = climul)
macro(cmulQ = climul)
macro(`&c` = climul)
macro(`&cQ` = climul)
macro(reversion = clirev)
macro(LC = LChig)
macro(RC = RCbig)
`Warning, new definitions for type/climon and type/clipolynom now include &C`
\-------------- End  ---------------
```
```
\-------------- Start  ---------------
\>\> There are no new global variables or macros in GTP yet. <<<
\-------------- End  ---------------
```
```
\> Global variables defined in Octonion:-setup are now available and have the se values: <<<
```
```
\-------------- Start  ---------------
_octbasis = [Id, e1, e2, e3, e4, e5, e6, e7]
_pureoctbasis = [e1, e2, e3, e4, e5, e6, e7]
_default_Fano_triples = [[1, 3, 7], [1, 2, 4], [1, 5, 6], [2, 3, 5], [2, 6, 7],
[3, 4, 6], [4, 5, 7]]
_default_squares = [-Id, -Id, -Id, -Id, -Id, -Id]
_default_Clifford_product = Clifford:-cmulNUM
\-------------- End  ---------------
```

See Also: `type/Fano_triples`, omultable, omul
(c) Copyright 1995 -- 2004, by Rafal Ablamowicz & Bertfried Fauser, all rights reserved.
Last revised: December 1, 2003 /RA