```
gap> LogTo("/home/irons/gapfiles/TraceCode.glog");
#   Already logging to /home/irons/gapfiles/TraceCode.glog
gap> F:=GF(4);


GF(2^2)

gap> FF:=GF(16);


GF(2^4)

gap> C:=RandomLinearCode(10,4,FF);


a  [10,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;


Variable: 'TraceCode' must have a value

0


brk>  ##inserted via emacs to test gap2latex.py
gap> CodeLength:= function (C)
> return WordLength(C);
> end;


function( C ) ... end

gap>
gap> TraceCode:= function (C,F)
> #C is defined over an extension of F
> #F is a base field
> local FF, TC, TrC, i, n, c;
> n:= CodeLength(C);
> FF:= LeftActingDomain(C);
> TC:= List(C,c->Codeword(List([1..n],i->Trace(FF,F,c[i]))));
> TrC:=ElementsCode(TC, F);
> IsLinearCode(TrC);
> return TrC;
> end;
```

```
function( C, F ) ... end

gap> TC:=TraceCode(C,F);time;


a linear [10,8,1]1..2 user defined unrestricted code over GF(4)
26072

gap> C:=RandomLinearCode(10,4,FF);


a  [10,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;


a linear [10,8,1..2]1..2 user defined unrestricted code over GF(4)
29663

gap> MinimumDistance(TC);


2

gap> MinimumDistance(C);


5

gap> C:=RandomLinearCode(10,4,FF);


a  [10,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;


a linear [10,8,1..2]1..2 user defined unrestricted code over GF(4)
32259

gap> MinimumDistance(C);


6

gap> MinimumDistance(TC);


2
```

```
gap> C:=RandomLinearCode(10,4,FF);

a  [10,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;

a linear [10,8,1..2]1..2 user defined unrestricted code over GF(4)
32021

gap> MinimumDistance(TC);

2

gap> MinimumDistance(C);

6

gap> C:=RandomLinearCode(10,4,FF);

a  [10,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;

a linear [10,8,1..2]1..2 user defined unrestricted code over GF(4)
34295

gap> MinimumDistance(C);

5

gap> MinimumDistance(TC);

2

gap> C:=RandomLinearCode(10,4,FF);

a  [10,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;
```

```
a linear [10,8,1..2]1..2 user defined unrestricted code over GF(4)
39789

gap> MinimumDistance(TC);


2

gap> MinimumDistance(C);


6

gap> C:=RandomLinearCode(12,4,FF);


a  [12,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;


a linear [12,8,1..3]2..4 user defined unrestricted code over GF(4)
33550

gap> MinimumDistance(C);


7

gap> MinimumDistance(TC);


3

gap> C:=RandomLinearCode(12,4,FF);


a  [12,4,?] randomly generated code over GF(16)

gap> TC
> ;


a linear [12,8,3]2..4 user defined unrestricted code over GF(4)

gap> TC:=TraceCode(C,F);time;
```

```
a linear [12,8,1..3]2..4 user defined unrestricted code over GF(4)
37376

gap> MinimumDistance(TC);


3

gap> MinimumDistance(C);


7

gap> C:=RandomLinearCode(12,4,FF);


a  [12,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;


a linear [12,8,1..3]2..4 user defined unrestricted code over GF(4)
38853

gap> MinimumDistance(C);


7

gap> MinimumDistance(TC);


3

gap> C:=RandomLinearCode(14,4,FF);


a  [14,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;


a linear [14,8,1..4]3..6 user defined unrestricted code over GF(4)
38631

gap> MinimumDistance(TC);


3
```

```
gap> MinimumDistance(C);

9

gap> C:=RandomLinearCode(14,4,FF);

a  [14,4,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;

a linear [14,8,1..3]3..6 user defined unrestricted code over GF(4)
37242

gap> MinimumDistance(C);

9

gap> MinimumDistance(TC);

3

gap> C:=RandomLinearCode(14,5,FF);

a  [14,5,?] randomly generated code over GF(16)

gap> C:=RandomLinearCode(12,5,FF);

a  [12,5,?] randomly generated code over GF(16)

gap> TC:=TraceCode(C,F);time;

exceeded the permitted memory (‘-o’ command line option) at
i := i + 1;
 called from
List( mat, ShallowCopy ) called from
BaseMat( VectorCodeword( AsSSortedList( C ) ) ) called from
IsLinearCode( TrC ); called from
<function>( <arguments> ) called from read-eval-loop
Entering break read-eval-print loop ...
you can ’quit;’ to quit to outer loop, or
you can ’return;’ to continue

brk>  gap>
gap>
gap>
```