~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

<exe-name> is a <version> host application.

Run with -? for options

-------------------------------------------------------------------------------

#748 - captures with unexpected exceptions

outside assertions

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

due to unexpected exception with messages:

answer := 42

expected exception

-------------------------------------------------------------------------------

#748 - captures with unexpected exceptions

inside REQUIRE\_NOTHROW

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

REQUIRE\_NOTHROW( thisThrows() )

due to unexpected exception with messages:

answer := 42

expected exception

-------------------------------------------------------------------------------

#835 -- errno should not be touched by Catch

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

CHECK( f() == 0 )

with expansion:

1 == 0

-------------------------------------------------------------------------------

'Not' checks that should fail

-------------------------------------------------------------------------------

ConditionTests.cpp:<line number>

...............................................................................

ConditionTests.cpp:<line number>: FAILED:

CHECK( false != false )

ConditionTests.cpp:<line number>: FAILED:

CHECK( true != true )

ConditionTests.cpp:<line number>: FAILED:

CHECK( !true )

with expansion:

false

ConditionTests.cpp:<line number>: FAILED:

CHECK\_FALSE( true )

with expansion:

!true

ConditionTests.cpp:<line number>: FAILED:

CHECK( !trueValue )

with expansion:

false

ConditionTests.cpp:<line number>: FAILED:

CHECK\_FALSE( trueValue )

with expansion:

!true

ConditionTests.cpp:<line number>: FAILED:

CHECK( !(1 == 1) )

with expansion:

false

ConditionTests.cpp:<line number>: FAILED:

CHECK\_FALSE( 1 == 1 )

-------------------------------------------------------------------------------

A METHOD\_AS\_TEST\_CASE based test run that fails

-------------------------------------------------------------------------------

ClassTests.cpp:<line number>

...............................................................................

ClassTests.cpp:<line number>: FAILED:

REQUIRE( s == "world" )

with expansion:

"hello" == "world"

-------------------------------------------------------------------------------

A TEST\_CASE\_METHOD based test run that fails

-------------------------------------------------------------------------------

ClassTests.cpp:<line number>

...............................................................................

ClassTests.cpp:<line number>: FAILED:

REQUIRE( m\_a == 2 )

with expansion:

1 == 2

-------------------------------------------------------------------------------

A couple of nested sections followed by a failure

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

explicitly with message:

to infinity and beyond

-------------------------------------------------------------------------------

A failing expression with a non streamable type is still captured

-------------------------------------------------------------------------------

TrickyTests.cpp:<line number>

...............................................................................

TrickyTests.cpp:<line number>: FAILED:

CHECK( &o1 == &o2 )

with expansion:

0x<hex digits> == 0x<hex digits>

TrickyTests.cpp:<line number>: FAILED:

CHECK( o1 == o2 )

with expansion:

{?} == {?}

-------------------------------------------------------------------------------

An unchecked exception reports the line of the last assertion

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

{Unknown expression after the reported line}

due to unexpected exception with message:

unexpected exception

-------------------------------------------------------------------------------

Contains string matcher

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( testStringForMatching(), Contains( "not there" ) )

with expansion:

"this string contains 'abc' as a substring" contains: "not there"

-------------------------------------------------------------------------------

Custom exceptions can be translated when testing for nothrow

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

REQUIRE\_NOTHROW( throwCustom() )

due to unexpected exception with message:

custom exception - not std

-------------------------------------------------------------------------------

Custom exceptions can be translated when testing for throwing as something else

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

REQUIRE\_THROWS\_AS( throwCustom(), std::exception )

due to unexpected exception with message:

custom exception - not std

-------------------------------------------------------------------------------

Custom std-exceptions can be custom translated

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

due to unexpected exception with message:

custom std exception

-------------------------------------------------------------------------------

EndsWith string matcher

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( testStringForMatching(), EndsWith( "this" ) )

with expansion:

"this string contains 'abc' as a substring" ends with: "this"

-------------------------------------------------------------------------------

Equality checks that should fail

-------------------------------------------------------------------------------

ConditionTests.cpp:<line number>

...............................................................................

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven == 6 )

with expansion:

7 == 6

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven == 8 )

with expansion:

7 == 8

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven == 0 )

with expansion:

7 == 0

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one == Approx( 9.11f ) )

with expansion:

9.1f == Approx( 9.1099996567 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one == Approx( 9.0f ) )

with expansion:

9.1f == Approx( 9.0 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one == Approx( 1 ) )

with expansion:

9.1f == Approx( 1.0 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one == Approx( 0 ) )

with expansion:

9.1f == Approx( 0.0 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.double\_pi == Approx( 3.1415 ) )

with expansion:

3.1415926535 == Approx( 3.1415 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello == "goodbye" )

with expansion:

"hello" == "goodbye"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello == "hell" )

with expansion:

"hello" == "hell"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello == "hello1" )

with expansion:

"hello" == "hello1"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello.size() == 6 )

with expansion:

5 == 6

ConditionTests.cpp:<line number>: FAILED:

CHECK( x == Approx( 1.301 ) )

with expansion:

1.3 == Approx( 1.301 )

-------------------------------------------------------------------------------

Equals string matcher

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( testStringForMatching(), Equals( "something else" ) )

with expansion:

"this string contains 'abc' as a substring" equals: "something else"

-------------------------------------------------------------------------------

Expected exceptions that don't throw or unexpected exceptions fail the test

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

CHECK\_THROWS\_AS( thisThrows(), std::string )

due to unexpected exception with message:

expected exception

ExceptionTests.cpp:<line number>: FAILED:

CHECK\_THROWS\_AS( thisDoesntThrow(), std::domain\_error )

because no exception was thrown where one was expected:

ExceptionTests.cpp:<line number>: FAILED:

CHECK\_NOTHROW( thisThrows() )

due to unexpected exception with message:

expected exception

-------------------------------------------------------------------------------

FAIL aborts the test

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

explicitly with message:

This is a failure

-------------------------------------------------------------------------------

FAIL does not require an argument

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

-------------------------------------------------------------------------------

FAIL\_CHECK does not abort the test

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

explicitly with message:

This is a failure

MessageTests.cpp:<line number>:

warning:

This message appears in the output

-------------------------------------------------------------------------------

INFO and WARN do not abort tests

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>:

warning:

this is a warning

-------------------------------------------------------------------------------

INFO gets logged on failure

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

REQUIRE( a == 1 )

with expansion:

2 == 1

with messages:

this message should be logged

so should this

-------------------------------------------------------------------------------

INFO gets logged on failure, even if captured before successful assertions

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

CHECK( a == 1 )

with expansion:

2 == 1

with messages:

this message may be logged later

this message should be logged

MessageTests.cpp:<line number>: FAILED:

CHECK( a == 0 )

with expansion:

2 == 0

with messages:

this message may be logged later

this message should be logged

and this, but later

-------------------------------------------------------------------------------

Inequality checks that should fail

-------------------------------------------------------------------------------

ConditionTests.cpp:<line number>

...............................................................................

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven != 7 )

with expansion:

7 != 7

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one != Approx( 9.1f ) )

with expansion:

9.1f != Approx( 9.1000003815 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.double\_pi != Approx( 3.1415926535 ) )

with expansion:

3.1415926535 != Approx( 3.1415926535 )

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello != "hello" )

with expansion:

"hello" != "hello"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello.size() != 5 )

with expansion:

5 != 5

-------------------------------------------------------------------------------

Matchers can be composed with both && and || - failing

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( testStringForMatching(), ( Contains( "string" ) || Contains( "different" ) ) && Contains( "random" ) )

with expansion:

"this string contains 'abc' as a substring" ( ( contains: "string" or

contains: "different" ) and contains: "random" )

-------------------------------------------------------------------------------

Matchers can be negated (Not) with the ! operator - failing

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( testStringForMatching(), !Contains( "substring" ) )

with expansion:

"this string contains 'abc' as a substring" not contains: "substring"

-------------------------------------------------------------------------------

Mismatching exception messages failing the test

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

REQUIRE\_THROWS\_WITH( thisThrows(), "should fail" )

with expansion:

expected exception

-------------------------------------------------------------------------------

Nice descriptive name

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>:

warning:

This one ran

-------------------------------------------------------------------------------

Non-std exceptions can be translated

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

due to unexpected exception with message:

custom exception

-------------------------------------------------------------------------------

Ordering comparison checks that should fail

-------------------------------------------------------------------------------

ConditionTests.cpp:<line number>

...............................................................................

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven > 7 )

with expansion:

7 > 7

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven < 7 )

with expansion:

7 < 7

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven > 8 )

with expansion:

7 > 8

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven < 6 )

with expansion:

7 < 6

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven < 0 )

with expansion:

7 < 0

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven < -1 )

with expansion:

7 < -1

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven >= 8 )

with expansion:

7 >= 8

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.int\_seven <= 6 )

with expansion:

7 <= 6

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one < 9 )

with expansion:

9.1f < 9

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one > 10 )

with expansion:

9.1f > 10

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.float\_nine\_point\_one > 9.2 )

with expansion:

9.1f > 9.2

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello > "hello" )

with expansion:

"hello" > "hello"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello < "hello" )

with expansion:

"hello" < "hello"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello > "hellp" )

with expansion:

"hello" > "hellp"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello > "z" )

with expansion:

"hello" > "z"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello < "hellm" )

with expansion:

"hello" < "hellm"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello < "a" )

with expansion:

"hello" < "a"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello >= "z" )

with expansion:

"hello" >= "z"

ConditionTests.cpp:<line number>: FAILED:

CHECK( data.str\_hello <= "a" )

with expansion:

"hello" <= "a"

-------------------------------------------------------------------------------

Output from all sections is reported

one

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

explicitly with message:

Message from section one

-------------------------------------------------------------------------------

Output from all sections is reported

two

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

explicitly with message:

Message from section two

-------------------------------------------------------------------------------

Pointers can be converted to strings

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>:

warning:

actual address of p: 0x<hex digits>

MessageTests.cpp:<line number>:

warning:

toString(p): 0x<hex digits>

-------------------------------------------------------------------------------

Reconstruction should be based on stringification: #914

-------------------------------------------------------------------------------

DecompositionTests.cpp:<line number>

...............................................................................

DecompositionTests.cpp:<line number>: FAILED:

CHECK( truthy(false) )

with expansion:

Hey, its truthy!

-------------------------------------------------------------------------------

SCOPED\_INFO is reset for each loop

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

REQUIRE( i < 10 )

with expansion:

10 < 10

with messages:

current counter 10

i := 10

A string sent directly to stdout

A string sent directly to stderr

Write to std::cerr

Write to std::clog

Interleaved writes to error streams

Message from section one

Message from section two

-------------------------------------------------------------------------------

StartsWith string matcher

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( testStringForMatching(), StartsWith( "string" ) )

with expansion:

"this string contains 'abc' as a substring" starts with: "string"

hello

hello

-------------------------------------------------------------------------------

Tabs and newlines show in output

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

CHECK( s1 == s2 )

with expansion:

"if ($b == 10) {

$a = 20;

}"

==

"if ($b == 10) {

$a = 20;

}

"

-------------------------------------------------------------------------------

Unexpected exceptions can be translated

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

due to unexpected exception with message:

3.14

-------------------------------------------------------------------------------

Vector matchers that fail

Contains (element)

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( v, VectorContains( -1 ) )

with expansion:

{ 1, 2, 3 } Contains: -1

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( empty, VectorContains( 1 ) )

with expansion:

{ } Contains: 1

-------------------------------------------------------------------------------

Vector matchers that fail

Contains (vector)

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( empty, Contains( v) )

with expansion:

{ } Contains: { 1, 2, 3 }

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( v, Contains( v2 ) )

with expansion:

{ 1, 2, 3 } Contains: { 1, 2, 4 }

-------------------------------------------------------------------------------

Vector matchers that fail

Equals

-------------------------------------------------------------------------------

MatchersTests.cpp:<line number>

...............................................................................

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( v, Equals( v2 ) )

with expansion:

{ 1, 2, 3 } Equals: { 1, 2 }

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( v2, Equals( v ) )

with expansion:

{ 1, 2 } Equals: { 1, 2, 3 }

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( empty, Equals( v ) )

with expansion:

{ } Equals: { 1, 2, 3 }

MatchersTests.cpp:<line number>: FAILED:

CHECK\_THAT( v, Equals( empty ) )

with expansion:

{ 1, 2, 3 } Equals: { }

-------------------------------------------------------------------------------

When unchecked exceptions are thrown directly they are always failures

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

due to unexpected exception with message:

unexpected exception

-------------------------------------------------------------------------------

When unchecked exceptions are thrown during a CHECK the test should continue

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

CHECK( thisThrows() == 0 )

due to unexpected exception with message:

expected exception

-------------------------------------------------------------------------------

When unchecked exceptions are thrown during a REQUIRE the test should abort

fail

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

REQUIRE( thisThrows() == 0 )

due to unexpected exception with message:

expected exception

-------------------------------------------------------------------------------

When unchecked exceptions are thrown from functions they are always failures

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

CHECK( thisThrows() == 0 )

due to unexpected exception with message:

expected exception

-------------------------------------------------------------------------------

When unchecked exceptions are thrown from sections they are always failures

section name

-------------------------------------------------------------------------------

ExceptionTests.cpp:<line number>

...............................................................................

ExceptionTests.cpp:<line number>: FAILED:

due to unexpected exception with message:

unexpected exception

-------------------------------------------------------------------------------

Where the LHS is not a simple value

-------------------------------------------------------------------------------

TrickyTests.cpp:<line number>

...............................................................................

TrickyTests.cpp:<line number>:

warning:

Uncomment the code in this test to check that it gives a sensible compiler

error

-------------------------------------------------------------------------------

Where there is more to the expression after the RHS

-------------------------------------------------------------------------------

TrickyTests.cpp:<line number>

...............................................................................

TrickyTests.cpp:<line number>:

warning:

Uncomment the code in this test to check that it gives a sensible compiler

error

-------------------------------------------------------------------------------

checkedElse, failing

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

CHECKED\_ELSE( flag )

with expansion:

false

MiscTests.cpp:<line number>: FAILED:

REQUIRE( testCheckedElse( false ) )

with expansion:

false

-------------------------------------------------------------------------------

checkedIf, failing

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

CHECKED\_IF( flag )

with expansion:

false

MiscTests.cpp:<line number>: FAILED:

REQUIRE( testCheckedIf( false ) )

with expansion:

false

spanner-------------------------------------------------------------------------------

just failure

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

explicitly with message:

Previous info should not be seen

-------------------------------------------------------------------------------

looped SECTION tests

s1

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

CHECK( b > a )

with expansion:

0 > 1

-------------------------------------------------------------------------------

looped tests

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

CHECK( ( fib[i] % 2 ) == 0 )

with expansion:

1 == 0

with message:

Testing if fib[0] (1) is even

MiscTests.cpp:<line number>: FAILED:

CHECK( ( fib[i] % 2 ) == 0 )

with expansion:

1 == 0

with message:

Testing if fib[1] (1) is even

MiscTests.cpp:<line number>: FAILED:

CHECK( ( fib[i] % 2 ) == 0 )

with expansion:

1 == 0

with message:

Testing if fib[3] (3) is even

MiscTests.cpp:<line number>: FAILED:

CHECK( ( fib[i] % 2 ) == 0 )

with expansion:

1 == 0

with message:

Testing if fib[4] (5) is even

MiscTests.cpp:<line number>: FAILED:

CHECK( ( fib[i] % 2 ) == 0 )

with expansion:

1 == 0

with message:

Testing if fib[6] (13) is even

MiscTests.cpp:<line number>: FAILED:

CHECK( ( fib[i] % 2 ) == 0 )

with expansion:

1 == 0

with message:

Testing if fib[7] (21) is even

-------------------------------------------------------------------------------

more nested SECTION tests

s1

s2

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

REQUIRE( a == b )

with expansion:

1 == 2

-------------------------------------------------------------------------------

send a single char to INFO

-------------------------------------------------------------------------------

MiscTests.cpp:<line number>

...............................................................................

MiscTests.cpp:<line number>: FAILED:

REQUIRE( false )

with message:

3

-------------------------------------------------------------------------------

sends information to INFO

-------------------------------------------------------------------------------

MessageTests.cpp:<line number>

...............................................................................

MessageTests.cpp:<line number>: FAILED:

REQUIRE( false )

with messages:

hi

i := 7

-------------------------------------------------------------------------------

string literals of different sizes can be compared

-------------------------------------------------------------------------------

TrickyTests.cpp:<line number>

...............................................................................

TrickyTests.cpp:<line number>: FAILED:

REQUIRE( std::string( "first" ) == "second" )

with expansion:

"first" == "second"

===============================================================================

test cases: 171 | 122 passed | 45 failed | 4 failed as expected

assertions: 980 | 871 passed | 88 failed | 21 failed as expected