Licensed under the Apache License, Version 2.0

```
1// Copyright 2012 Google
...
23#include <vector>
24
25#include "base/logging.h"
26#include "constraint_solver/constraint_solver.h"
27
28namespace operations_research {
29
30// helper functions
31IntVar* const MakeBaseLine2(Solver*  s,
...
36}
37
38IntVar* const MakeBaseLine3(Solver* s,
...
53}
54
55IntVar* const MakeBaseLine4(Solver* s,
...
73}
74
75void CPIsFun() {
76  // Constraint programming engine
77  Solver solver("CP is fun!");
78
79  const int64 kBase = 10;
80
81  // Decision variables
82  IntVar* const c = solver.MakeIntVar(1, kBase - 1, "C");
...
91  IntVar* const e = solver.MakeIntVar(0, kBase - 1, "E");
92
93  // We need to group variables in a vector to be able to use
94  // the global constraint AllDifferent
95  std::vector<IntVar*> letters;
96  letters.push_back(c);
...
105  letters.push_back(e);
106
107  // Check if we have enough digits
108  CHECK_GE(kBase, letters.size());
109
110  // Constraints
111  solver.AddConstraint(solver.MakeAllDifferent(letters       ));
112
113  // CP + IS + FUN = TRUE
114  IntVar* const term1 = MakeBaseLine2(&solver, c, p, kBase);
115  IntVar* const term2 = MakeBaseLine2(&solver, i, s, kBase);
116  IntVar* const term3 = MakeBaseLine3(&solver, f, u, n, kBase);
117  IntVar* const sum_terms = solver.MakeSum(solver.MakeSum(term1,
118                                                          term2),
119                                           term3)->Var();
120
121  IntVar* const sum   = MakeBaseLine4(&solver, t, r, u, e, kBase);
122
123  solver.AddConstraint(solver.MakeEquality(sum_terms, sum));
```

Headers

Namespace

Helper functions

Is it?

CP solver

Variables

Assert-like macro

Constraints