```
// $Id: attributes.cc,v 1.1 2013-09-24 18:51:15-07 - - $

//
// Example which shows how to manage attributes as bitsets and
// how to print them.
//

#include <limits.h>
#include <stdio.h>
#include <stdlib.h>

typedef unsigned long bitset_t;

typedef enum {FALSE = 0, TRUE = 1} bool;

enum {
   ATTR_INDEX_VOID     =  0,
   ATTR_INDEX_BOOL     =  1,
   ATTR_INDEX_CHAR     =  2,
   ATTR_INDEX_INT      =  3,
   ATTR_INDEX_NULL     =  4,
   ATTR_INDEX_STRING   =  5,
   ATTR_INDEX_STRUCT   =  6,
   ATTR_INDEX_ARRAY    =  7,
   ATTR_INDEX_FUNCTION =  8,
   ATTR_INDEX_VARIABLE =  9,
   ATTR_INDEX_FIELD    = 10,
   ATTR_INDEX_TYPEID   = 11,
   ATTR_INDEX_PARAM    = 12,
   ATTR_INDEX_LVALUE   = 13,
   ATTR_INDEX_CONST    = 14,
   ATTR_INDEX_VREG     = 15,
   ATTR_INDEX_VADDR    = 16,
};

const bitset_t ATTR_VOID     = 1 << ATTR_INDEX_VOID;
const bitset_t ATTR_BOOL     = 1 << ATTR_INDEX_BOOL;
const bitset_t ATTR_CHAR     = 1 << ATTR_INDEX_CHAR;
const bitset_t ATTR_INT      = 1 << ATTR_INDEX_INT;
const bitset_t ATTR_NULL     = 1 << ATTR_INDEX_NULL;
const bitset_t ATTR_STRING   = 1 << ATTR_INDEX_STRING;
const bitset_t ATTR_STRUCT   = 1 << ATTR_INDEX_STRUCT;
const bitset_t ATTR_ARRAY    = 1 << ATTR_INDEX_ARRAY;
const bitset_t ATTR_FUNCTION = 1 << ATTR_INDEX_FUNCTION;
const bitset_t ATTR_VARIABLE = 1 << ATTR_INDEX_VARIABLE;
const bitset_t ATTR_FIELD    = 1 << ATTR_INDEX_FIELD;
const bitset_t ATTR_TYPEID   = 1 << ATTR_INDEX_TYPEID;
const bitset_t ATTR_PARAM    = 1 << ATTR_INDEX_PARAM;
const bitset_t ATTR_LVALUE   = 1 << ATTR_INDEX_LVALUE;
const bitset_t ATTR_CONST    = 1 << ATTR_INDEX_CONST;
const bitset_t ATTR_VREG     = 1 << ATTR_INDEX_VREG;
const bitset_t ATTR_VADDR    = 1 << ATTR_INDEX_VADDR;

bitset_t bitset (int attribute_index) {
   return 1L << attribute_index;
}

```

```

bool is_primitive (bitset_t attributes) {
   return attributes
          & (ATTR_BOOL | ATTR_CHAR | ATTR_INT)
       && ! (attributes | ATTR_ARRAY);
}

bool is_reference (bitset_t attributes) {
   return attributes
          & (ATTR_NULL | ATTR_STRING | ATTR_STRUCT | ATTR_ARRAY)
       && TRUE;
}

// The following initialization style is a gcc-ism and will
// not work with some C compilers, and confuses lint.
const char *attr_names[] = {
   [ATTR_INDEX_VOID    ] "void"    ,
   [ATTR_INDEX_BOOL    ] "bool"    ,
   [ATTR_INDEX_CHAR    ] "char"    ,
   [ATTR_INDEX_INT     ] "int"     ,
   [ATTR_INDEX_NULL    ] "null"    ,
   [ATTR_INDEX_STRING  ] "string"  ,
   [ATTR_INDEX_STRUCT  ] "struct"  ,
   [ATTR_INDEX_ARRAY   ] "array"   ,
   [ATTR_INDEX_FUNCTION] "function",
   [ATTR_INDEX_VARIABLE] "variable",
   [ATTR_INDEX_FIELD   ] "field"   ,
   [ATTR_INDEX_TYPEID  ] "typeid"  ,
   [ATTR_INDEX_PARAM   ] "param"   ,
   [ATTR_INDEX_LVALUE  ] "lvalue"  ,
   [ATTR_INDEX_CONST   ] "const"   ,
   [ATTR_INDEX_VREG    ] "vreg"    ,
   [ATTR_INDEX_VADDR   ] "vaddr"   ,
};

void print_attributes (bitset_t attributes) {
   ssize_t size = sizeof attr_names / sizeof *attr_names;
   for (int index = 0; index < size; ++index) {
      if (attributes & bitset (index)) {
         printf (" %s", attr_names [index]);
      }
   }
}

int main (void) {
   printf ("Number of bits in a bitset = %lu\n",
           CHAR_BIT * sizeof (bitset_t));
   for (bitset_t set = 0xF; set < 1L << 32; set <<= 4) {
      printf ("bitset 0x%016lX =", set);
      print_attributes (set);
      printf ("\n");
   }
   return EXIT_SUCCESS;
}

/*
//TEST// attributes >attributes.out 2>&1
//TEST// mkpspdf attributes.ps //TEST//    attributes.cc* attributes.out
*/

```

```
* @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting attributes.cc
* attributes.cc: $Id: attributes.cc,v 1.1 2013-09-24 18:51:15-07 - - $
* g++ -g -O0 -Wall -Wextra -std=gnu++0x attributes.cc -o attributes -lm
```

```
Number of bits in a bitset = 64
bitset 0x000000000000000F = void bool char int
bitset 0x00000000000000F0 = null string struct array
bitset 0x0000000000000F00 = function variable field typeid
bitset 0x000000000000F000 = param lvalue const vreg
bitset 0x00000000000F0000 = vaddr
bitset 0x0000000000F00000 =
bitset 0x000000000F000000 =
bitset 0x00000000F0000000 =
```