cmp — at most 190 points
(the Scientific Committee only has a solution for 100 points)

Source code: **cmp.c, cmp.cpp, cmp.pas**

Time limit: **10 seconds**

Memory limit: **16 MB in addition to what Committee's library uses**

You must design an algorithm for an ancient computer whose memory is just an array of 10240 bits. The memory is initialized to zero, after which you may write and read a single bit at a time.

You must implement two operations on this computer:

remember(*a*): *a* is an integer between 0 and 4095
The implementation of this operation may call:
bit_set(*address*): *address* is an integer between 1 and 10240
The memory bit at position *address* will be set to 1

compare(*b*): *b* is an integer between 0 and 4095
if *b* < *a*, it should return -1
if *b* = *a*, it should return 0
if *b* > *a*, it should return 1
The implementation of this operation may call:
bit_get(*address*)
Returns the memory bit at position *address*: 1 if it was set through *bit_set*() during the *remember*(a) operation, and 0 otherwise.

Task

Implement *remember*() and *compare*() to minimize the total number of memory accesses (*bit_set*() and *bit_get*()) in the worst-case for all possible values of *a* and *b*.

Your solution will be evaluated exhaustively:
define AllMemory = array [0..4095][1..10240] of bits
set AllMemory to zeros
for *a* = 0..4095:
define *bit_set*(*address*): AllMemory[*a*][*address*] = 1
remember(*a*)
let *maxA*= the maximum number of *bit_set*() calls executed for any *a*
for (*a*,*b*) ∈ {0..4095}×{0..4095} in random order (i.e. all valid pairs (a,b) are considered, in some random order)
define *bit_get*(*address*): return AllMemory[*a*][*address*]
answer =*compare*(*b*)
if *answer* for comparing *a* and *b* is incorrect : your score = 0; exit
let *maxB* = the maximum number of *bit_get()* calls executed for any (*a*,*b*) pair
T=*maxA* + *maxB*
If (*T*>20): your score = 0; exit
else your score = 1 + 9 * (21– *T*); exit

Description of implementation

Implementation in C:

Your source file must begin with:

```
#include "cmp.h"
```

The prototypes for the memory-access functions are:

```
void bit_set(int addr);
int bit_get (int addr);
```

You must implement the functions:

```
void remember(int value);
int compare(int value);
```

The file *cmp.h* will provide the function *main*(), and you must not define another such function. All global names (for variables and functions) except *bit_set*() and *bit_get*() will begin with the prefix *boi*. So you may avoid compilation errors by not naming your functions and variables with this prefix.

We have provided a sample implementation for *cmp.h* in the file *public-cmp.h* for convenience which you may edit as you please. The implementation is different from the version of *cmp.h* used for grading but the interface to your program remains the same.

To compile all code simply compile your source file and make sure that *cmp.h* is in the same directory.

Implementation in Pascal:

You must implement a unit with the name *cmp* in the file cmp.pas. Your program will use the *bit_set*() procedure and the *bit_get*() function defined in the *cmpdata* unit, and having the prototypes:

```
procedure bit_set(addr:integer);
function bit_get(addr:integer):integer;
```

Your unit must implement the *remember()* procedure and *compare()* function which will be called by the main program.

```
procedure remember(value:integer);
function compare(value:integer):integer;
```

The main program will be provided by the Scientific Committee during evaluation. For your convenience, we have provided a sample implementation in the file *public-cmp.pas* (which you may edit if you want). This is different from the implementation used during evaluation. If you place *cmpdata.pas*, *cmp.pas* and *public-cmp.pas* in the same directory, you can compile all code using:

```
fpc public-cmp.pas
```

The code we provide (cmpdata.pas and the main program) only declares variables prefixed with the name *boi*. You must not attempt to access these variables and to prevent name clashes and compilation errors, you should avoid using the *boi* prefix for your own variables and functions.

Downloading files:

Please access the Web page for this problem and click on "*download task description*" to obtain the *cmp.zip* archive. This archive contains *cmp.h*, *cmpdata.pas*, and *public-cmp.pas*, which you may use for developing your solution, the problem statement and sample implementation sources *cmp.pas*, *cmp.c*, and *cmp.cpp*.

Constraints

- You may be disqualified for any attempt to interact with the memory of our grading code.
- If your solution does not obey the protocol defined above (e.g. calling *bit_set*() during *compare*() or with an invalid address), you will receive zero points.
- **Time limit:** your implementation must execute 4096 calls to *remember*() and 4096*4096 calls to *compare*() in under 10 seconds.