THREE BLIND MICE (16K min.). TRS-80 or Video-Genie

MICE is a Level2 BASIC Program which has been translated into Z80 machine-code by Southern Software's ACCEL2 compiler, to get better performance. The compiled version is on the labelled side of the tape. Use this version to play the game. On the unlabelled side of the tape is the original BASIC source program. You can CLOAD this and RUN it, but it is about 20 tines slower than the compiled version, and this makes the game quite pointless.

LOADING THE PROGRAM.

The labelled side of the cassette holds a core-image file, which will set MEMORY SIZE automatically. Under Level2, (not Disk BASIC), type:

```
SYSTEM    (enter)
*? MICE   (enter)             This is the file name
...                           The tape loads
*? /      (enter)
READY
```

The tape load is just like any other TRS-80 tape load, and subject to the same vagaries. You may have to adjust your volume setting to match the characteristics of the tape supplied. Contrary to normal Southern Software practice, only one copy of each program is provided (to reduce duplication time). If this copy is unloadable, then return the cassette for an immediate replacement. But please make every reasonable effort to load the tape first.

RUNNING THE PROGRAM.

MICE runs like any other BASIC program. Type:

```
RUN               (enter)
LENGTH OF TAIL? 15 (enter)
Game starts.
```

You can set the length of the tails of the mice to any number between 1 and 15. The game is easier if the tails are longer. On the screen you will see three "mice" with big fat heads and long tails. They move at random without intelligence. In the middle of the screen is the "Farmer's Wife", the small dot. She can be moved by pressing any of the four arrowed keys, up, down, left, or right. (On Video-Genie you can use the keys Y,B,G, and H instead). The Farmer's Wife cannot move faster than the mice. She dies, and the mice win the game, if she gets run over by the head of one of the mice. Your aim, by guiding the Farmer's Wife, is to "cut-off" (i.e. run over) the tails of all three mice, while avoiding their heads. When one mouse dies, the others move faster, and the last is the fastest and most difficult to catch.

That's all there is to running the program. The rest of this description relates to the merits of compilation, and the characteristics of the ACCEL compilers.

COMPARATIVE STATISTICS ON MICE PROGRAM.

Times quoted are for 40 cycles of the program, i.e. 40 movements of the mice with no keyboard activity.

MICE	Time	Size
Uncompiled	117 secs.	3642 bytes.
Compiled	6 secs.	4458 bytes.

So, for this program, compilation has yielded a speed improvement of nearly 20, against a space degradation of about a quarter. These ratios vary widely according to the type of program. If your source programs contain no REMarks, then you should expect a higher code expansion factor.

southern
software

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

FORMAT OF THE COMPILED PROGRAM.

If you LIST the loaded program you will see that much of it consists of null REMarks. These are the original BASIC lines converted to machine-code. You cannot EDIT the program, or modify it in any way. If you attempt to, then, because these lines contain bytes that may, by chance, be treated as special control codes, you may cause a MEMORY SIZE restart. (You can clear out the program with NEW or CLOAD). It is generally true of compiled code that it cannot be modified. Instead, you have to reload the original source, modify that, and then recompile, which of course you cannot do unless you have the ACCEL2 compiler.

You may very well believe that you could create a much better game than MICE, or you may have ideas for other, more valuable, programs in BASIC that you could sell, if only they would run faster. This is what a compiler is for, and selling the compiled program has the side-effect that the program is protected from modification. Only the originator has the source, and only he can make changes or corrections. This discourages program piracy.

THE UNCOMPILED PROGRAM.

Compare the compiled listing with the uncompiled, by loading the BASIC program on the flip side with CLOAD. A novel feature of the ACCEL2 compiler is that it produces programs that run under the BASIC environment, using the BASIC variables. An I/O statement, such as C$=INKEY$ at line 2000 is not translated, but is left in source BASIC. This keeps the compiler relatively small and simple. In addition, ACCEL2 has an option which lets the user minimise translation of a section of the program. Typically the segment would be used for initialisation, or exception handling, and would not be critical to performance. The option REM NOEXPR tells ACCEL2 not to translate expressions, while REM EXPR turns translation back on. This is used at 990, around initialisation, at 3001, when a hit on the Farmer's Wife has been detected, and at 5990 when a mouse has been killed. This selective non-translation has no detrimental effect on the game, but it reduces the compiled program size from 5279 to 4458 bytes.

Like most games, MICE requires random numbers. When the RND(n) function was used in line 4000, the run speed deteriorated by 50%. (RND is a particularly slow function.) So, instead, random numbers are generated by using PEEK on consecutive bytes of ROM. This "problem" is not an effect of compilation, per se, but it illustrates that when a program is compiled, "trivial" non-compilable operations may come to dominate the run speed.

CONFIGURATION CHOICES.

This sample version of MICE is made available only in one configuration, namely, loaded from cassette, running on Level2, and in 16K (or more). This is the "lowest common denominator" system under which MICE will run. It is not a choice forced by the compiler.

1) Level2 or Disk BASIC. Accel2 will run under Level2 or Disk BASIC, and will accept programs in the full Disk BASIC language, (e.g. disk I/O, hex constants, DEF FN, etc.)

2) Tape or disk. ACCEL2 provides routines for saving and loading compiled programs to and from disk, and it is possible to "chain" compiled programs running from disk. To save programs compiled by ACCEL or ACCEL2 on tape, you need the separate Southern Software utility TSAVE, (used to create the enclosed tape).

3) Memory Size. Compiled code is generally larger than source, and the compiler itself occupies space, so you need spare memory above that required by BASIC to run the original. Bearing this in mind, ACCEL and ACCEL2 will both run under Level2 in 16K or more. ACCEL2 will run under Disk BASIC in 32K or more.

4) ACCEL or ACCEL2. For a disk-based system ACCEL2 is necessary. For non-disk, ACCEL and ACCEL2 produce equal performance improvements for those functions they both translate to machine-code. However ACCEL does not optimise array references, nor expressions involving data-types other than INTEGERs. Generally speaking, games programs can be handled very well with ACCEL, especially if you are willing to use PEEK and POKE to replace arrays. But for numerical programs, or string handling, then ACCEL2 is necessary. Neither will make any impression on programs entirely limited by I/O. If in doubt, remember that you can purchase ACCEL and then replace it by ACCEL2 without loss.