XYZZYnews

January/
February 1995
Issue #1

The Magazine for Interactive Fiction Enthusiasts

HOLLOW VOICE

You pick up the magazine. As you flip through its pages, an editorial catches your eye.

```
>EXAMINE EDITORIAL
```

Hello! And welcome to the premier issue of *XYZZYnews: The Magazine for Interactive Fiction Enthusiasts*. With this 'zine, I hope to create an open forum for fellow gamers who are crazy about computer adventure games, especially text-based adventures. This includes nostalgic fans of the old Infocom games, people who're always on the lookout for new games to play, and designers of new adventure games. *XYZZYnews* is for anyone who favors computer games that compel players to face intellectual challenges or a series of logic puzzles in order to complete a storyline.

Longtime IF fans will recognize the reference to "xyzzy" as the magic word in the original "Advent" or "Adventure" that transports the player from anywhere in the game to a central location. Continuing the "Adventure" theme, you'll notice that the title of this editorial column is "Hollow Voice". If you type "xyzzy" in many text adventures (presumably to see if that strategy will work here too), a "hollow voice" derides your efforts. In contrast, this Hollow Voice won't laugh at anyone's attempts; I hope only that my 'zine and I can be resources for exploring all avenues of adventure gameplay.

In the pages of this issue you'll find plenty of game reviews; an interview with Graham Nelson, game designer extraordinaire and developer of the Inform language; a sneak preview of some upcoming text adventures; a how-to piece for game writers on avoiding 10 common design mistakes; the first installment of the Spoiler Column, featuring hints or answers to questions that've come into our e-mailbox; and a step-by-step look at how one game designer solved a sticky bug in his code.

I hope that this first issue proves entertaining, informative, and interesting to you. And naturally, I'm looking for your feedback, more game reviews, articles, walkthroughs, Q&A, and other ideas for the second issue. Please write or email me with any feedback or suggestions you may have.

Happy gaming!

Eileen Mullin
xyzzynews@aol.com

Contents:

XYZZY*news*

XYZZYnews is published bimonthly by Bran Muffin Communications, 320 West 84th Street, Ste. 5B, New York, NY 10024, USA. Email: *xyzzynews*@aol.com. Send all inquiries, letters, and submissions to the address above.

Contents © 1995 *XYZZYnews*. All rights reserved. Published in the United States of America.

Electronic versions: There are currently two versions of *XYZZYnews* made available online. One is in ASCII and can be viewed with any text reader. You can also download a .PDF file that mirrors the layout of the print version. Use the Adobe Acrobat Reader (available for Windows, Mac, DOS and UNIX) to view the .PDF file; no special fonts or linked graphics are needed. You can obtain Acrobat Reader at ftp.adobe.com or http://www.adobe.com.

Subscriptions: Both electronic versions are available at no cost. You can obtain either one by FTPing to ftp.gmd.de. To be added to the mailing list, please write to xyzzynews@aol.com and specify text-only or .PDF version. The print version includes a 3.5" Mac or PC disk and is $21 (U.S.) for one year (6 issues) or $3.50 for a sample issue. For print subscriptions outside the U.S. or Canada, please email or write for rates.

All products, names, and services are trademarks or registered trademarks of their respective companies.

Editorial deadline for Issue #2 is February 28, 1995.

Editor:
Eileen Mullin

Contributors to this issue:
Stuart Beach
Neil deMause
C.E. Forman
Melissa Katz
Lauren Meckler
Conrad Wong

Sneak Previews

This issue, “Sneak Previews” looks at three TADS games that are all set in academic environments. Since all are still in development, their release date information may change.

The Legend Lives! by David Baggett, distributed by Adventions, is the highly-anticipated next installment in the Unnkulian series. As a graduate student at Akmi Yooniversity, you’ve stumbled across a terrible discovery while researching your thesis—the code to an Unnkulian-created virus that’s been set loose on the intergalactic AkNet computer network. You need to get to the bottom of several mysteries: figuring out how to stop the virus, determining if the Akmi company has once again joined the Unnkulians’ dark forces, and finding out where the Unnkulians have been hiding and what they’ve been doing in recent years.

“The Legend Lives!” has a novel method for changing locations; you must enter a matter mover to transport from one place to another, and you must dial the address combination for your desired destination (and have permission to transport if it’s a private address) before you can go anywhere; there’s little aimless wandering in this game. As in earlier Unnkulian games, you’ll find plenty of cheezy product placements for Akmi goods. Online hints are also available. Look for “The Legend Lives” to release sometime this spring.

MacWesleyan: An Everyday Nightmare (Mac version; the PC version is called **PC University**) by Neil deMause is a surreal exercise in academic bureaucracy. Your goal is to obtain all the necessary signatures on your Student Identification Form and return it to the Registrar’s office. This includes getting signatures from your faculty advisor, department chair, dean, the university president, and the U.S. president, not to mention getting your own signature (harder than it sounds). Based at Wesleyan University in Middletown, CT, the game also features speeding Domino’s pizza trucks that hinder your safe passage, unusual psychology experiments, and a bus labeled “House of God” that indeed houses the Famous One. “MacWesleyan” is expected to release by the end of January.

John’s Fire Witch by John Baker opens at Ohio State University, where you’re on your way to hang out with John Baker himself for an evening of consuming major amounts of pizza and beer. He’s a no-show, so you head back to his place and crash for the night, only to awaken to a raging blizzard outside and still no sign of John. As the game continues, you learn John is wrapped up in a struggle between a “Fire Witch” and an “Ice Wizard” that live in a recently exposed underground area far below his Columbus apartment. “John’s Fire Witch” is billed by its author as “a good beginner game”; it currently has an expected release date of February 1.

Interview: Graham Nelson

Graham Nelson has intrigued thousands of text adventure fans with his enchanting game "Curses", among others. He is also well-known as the author of Inform, a freely distributed, highly portable development system for writing text adventures, and has been involved in advising others on making use of it.

Q. ***How did you first become interested in adventure game programming, and how and when did you get started? What kind of computer do you use, and what programming language(s) have you used in the past?***

I'm a rather junior pure mathematician at Oxford University, aged 27: I mention my age because I was a child when Crowther and Woods' original "Advent" was being circulated across in the world in the late 1970s. My next-door neighbor, a senior executive at Digital UK, occasionally took me to play at weekends, in a hugely untidy maze of a converted biscuit factory in Reading. I was mesmerized, about 10 being the peak age for imagination. Then in my early teens, when the home computer bubble was blowing, I had one of the first, an Acorn Atom, and used to write primitive adventures on that. (In about 10K of memory, which taught me a lot.) The breakthrough was realizing that one could represent locations with numbers, giving each object a location variable...well, I was young. I still use Acorn machines, which are technically superb: I now have an Acorn Archimedes A5000 (which contains the old Atom operating system the way human DNA still includes that of early grasses). I use two languages, the excellent Norcroft ANSI C compiler and Inform.

Q. ***What was the chronology of events in your development of Inform and Curses? How long did the different parts of Curses' game development and programming processes take?***

Firstly, I never intended to write Inform! I had played a few Infocom games in the late 1980s (running under CP/M on a ghastly Amstrad), and admired them: I was curious about the run-time format and in early 1993 found some fragmentary documents on the Internet. I wrote to Mark Howell asking if there was a compiler, and he said "Many people have had many dreams". So I tried to fake a toy game file, to print "Hello World" and stop. Each time, nothing happened, and I thought, ah, I shouldn't have missed out the property default table, and coded a bit more. Eventually I found it had been working all along—but didn't show anything on screen until it had the first full page of text. I inserted 30 new lines, and suddenly my toy said "hEllO woRlD". (An hour later I understood alphabet shifting rather better!) My first real program factorized the numbers 2 to 100 into prime factors, for which I improvised an assembler: Mark's excellent disassembler helped me to perfect it. By then "zass", as it was called, was a horrid tangle of exceptions so I sat down with plenty of paper and worked out the underlying rules. I wrote these into a much simpler assembler, added "if" statements, code blocks, loops and an expression evaluator, and the result was Inform 1.

The first large Inform program was the hardest one, the parser. I was determined on all the Infocom trappings: asking questions to sort out ambiguities, in particular.

Adding everyday Adventure verbs (take, drop, inventory) was simple: finally I threw away my silly test game and began “Curses”. I had a rough design in mind already, and in about a fortnight had the attics, the Unreal City and the garden fleshed out. I should really have planned more carefully: for example, the game’s least fair puzzle (changing the torch battery) dates from then. Some friends amused themselves finding a bug every ten seconds: I took it away, kept at it for a month or so over Easter 1993 and gave it to Richard Tucker and Gareth Rees to test. They sent me back another hundred bugs, and I filled out the game with a few more puzzles (such as the robot mouse). At the end of April I archived “Curses” and Inform, and announced them on the newsgroups. For a fortnight nobody at all emailed me, or posted a follow-up. Doesn’t anyone care, I thought? It turned out my newsreader was broken, and hadn’t posted at all. Once the announcements were actually heard, there was a slow but gathering response: by the end of the first year, an avalanche.

The first public “Curses” (release 7) was a little over half the present size: the catacombs, Alexandria and the village were missing, for instance, and these were gradually added in later releases. (It’s now finished. The very last addition was the maker’s mark of my 18th century ancestor, William Snelson the Clockmaker, on the parish clock.) Extensions were possible because Inform was getting better, with the help of real programmers, especially Bob Newell and Dilip Sequeira. By the new year of 1994, it had grown up into Inform 4 and could produce games twice as large. The language was still difficult to write games in, though (I wince when I look at the original “Curses” source code). I decided on a makeover and did this by writing an Inform version of “Advent”, working from David Baggett’s excellent TADS edition: on the way I improved the cosmetics of Inform until it looked object-oriented and comprehensible. In the nine months since, I’ve kept the language stable (though the parser has vastly improved) and worked instead on proper documentation: I’m rather pleased with the new manuals. I see Inform now as a gauche young adult, having got past the stage of growing out of his shoes every few months.

Q. ***You’ve chosen to distribute Inform, as well as your games, free of charge. What was your line of reasoning in making this altruistic decision?***

At the time the newsgroups had a rather elegiac “all the good games were written years ago” tone, and I hope “Curses” has contributed to a change in that attitude. I wanted to revive the “dead format” of the Infocom game, and persuading people is easier when they don’t have to pay to listen to you. In the early Renaissance, Italian artists would wander round Roman ruins and say, well, we can build arches too…and I’m as vain as those artists. There are other kinds of profit.

Q. ***Have you seen any interesting games that others have created using Inform?***

Yes, but mostly ports from earlier systems, fragments and work in progress (I know about a dozen authors working seriously, and I do my best to help them with technical snags): Inform 5 is still rather new. I do recommend Jon Drukman’s witty “Busted”, since you ask.

Q. ***Over the past few months, questions about Curses (and pleas for hints) have virtually precluded all other discussion on the interactive fiction newsgroups. What comments/questions do you hear most frequently? Any extremely odd complaints?***

Remember that "Curses", being free, is circulated much more widely than shareware games (it has reached tens of thousands of players, the approximate sales level of a typical Infocom game at their height), so it gets more than its fair share of attention. It's frequently over-praised now, and I've enjoyed watching a generation of "angry young men" critics beginning to say, well what's so good?

The most frequent complaint is that it's hard. True: it's a hard game to win (though surprising numbers of people do, and I tried to put much of the best material in the late middle game, to reward persistence). Also, many people ask me how to use the secret debugging commands, apparently under the impression that I'll tell them.

Q. ***In retrospect, what would you have done differently in designing Curses?***

I like the eclectic feel of the game, the wide range of puzzles: so I'd keep that, even though it looks incoherent in places (especially to people who miss the better-hidden clues). I might make it a little smaller: as it is, there was a draft which had a good 20 or so extra rooms, but my play-tester Michael Kinyon and I never felt happy about them and I removed two entire regions. (Redrafting heavily after testing, incidentally, is much to be recommended: testing isn't just about bug-fixes.) What I would pay much more attention to are the few points where the player can inadvertently make a career decision. Most players end up back-tracking, though some actually enjoy this.

Q. ***Please describe the demonstration files that showcase Inform's features.***

First, "Advent", a simple game to implement but still a good one to play. Then there's "Toyshop", a dense little area filled with peculiar objects: examples for expert Informers. "Balances"—which I wrote in one and a half evenings, and isn't meant to be a serious work of art!—is a pastiche of the much-loved "Enchanter" trilogy, partly (I admit) for fun but mostly to demonstrate really awkward "parser" puzzles, such as lottery tickets numbered from 1 to 10,000, and indistinguishable objects which one can write names on. There are also some smaller examples, and a port of Scott Adams' "Adventureland".

Q. ***There has been heated discussion on the IF newsgroups about the difficulty level and fairness of some of the puzzles in Balances and Curses. How do you respond to charges that some of your puzzles violate tenets of your own "Bill of Player's Rights"?***

Guilty. But in mitigation I do mostly play fair, and am at least properly contrite about the few low blows in "Curses". I try to make puzzles range all the way from easy to hard (most are never complained about), and to leave many open at once. A deliberate choice on my part was for the player to continue to find new possibilities in the early Attic rooms far into the game: I think this builds atmosphere, though it means there's no neat division of the prologue from the middle game. Besides, players very widely disagree (especially with me) about what's hard and what's easy: and in a way, "I won, but it was a fight" is the best compliment a game can receive.

Q. ***What plans do you have for future games? Is a Curses sequel a possibility?***

I have been working on a more serious game, called “Jigsaw”, for about 18 months (off and on), but don’t hold your breath: it’ll be a while arriving. The time has mainly gone on getting Inform into a decent shape for public use. I suppose the plot of “Curses” makes a sequel conceivable (when compared with, say, the plot of “Hamlet”) but none is planned.

Q. ***Who are your favorite authors? And what are your favorite computer games?***

I greatly admire the poetry of Philip Larkin and Primo Levi, writers who have absolutely nothing in common but their initials. Besides them, Auden, Eliot, Donne, Browning, Elizabeth Bishop: more predictable choices. For plays, Tom Stoppard, Christopher Hampton, David Hare. My favorite novel, I admit it, is John Christopher’s trilogy “The White Mountains”, “The City of Gold and Lead” and “The Pool of Fire”: notionally for (older) children, but infinitely more sophisticated than, say, Tolkien. His prose is not flawless but his plot, characterization and atmosphere are.

In my view the best adventure games are “Spellbreaker” and “Trinity”, coming up for its tenth anniversary. The modern computer games market depresses me: I think it’s become like American TV. Nowadays, an “adventure game” means something like “The Seventh Guest”, on two CD-ROMs, about 20,000 times as much code as the original “Advent” and about a tenth as clever or interesting. “Alone In The Dark” is better, admittedly. I had a serious “Tetris” problem at one time. Shoot-em-up games are all very well, but I prefer them abstract—old-fashioned as I am, I find reviews like “I saw tortured POW’s, floating (dead) comrades…Great stuff!” (a genuine example) rather disturbing.

Q. ***I’ve heard a lot about how portable Inform is to various systems because it works with so many interpreters. Would you elaborate?***

The compiler itself will indeed run on most sensible computers (such as PCs and Macs) but the games it produces will run on practically anything, from personal organizers to mainframes. An “interpreter” is a program which will play an Infocom game file, whether bought from Activision (who currently sell the Infocom back catalogue at very low prices) or compiled by Inform. There are public-domain interpreters for virtually every machine (and it requires little effort to move them to new ones). So there’s only one “Curses” program, and it runs identically on dozens of very different machines.

Q. ***What is your view on future directions for interactive fiction?***

I don’t really believe in “directions” in art; the rope twists as you follow it, that’s all. The “interactive fiction” format hasn’t changed in any fundamental way since the early 1970s, in the same way that the format of the novel hasn’t since 1700. (Don’t believe the cyberpunks—the majority of authors aren’t going to move over to hypertext just because the technology now exists, any more than they did to pop-up books, comics or television.) If pushed, though, I’d say that the next stage will be reached when it it’s no longer true that about 75% of the best games were written in 1980–85. We’re on the way to that.

Q. ***What other work/projects currently occupy your time? What do you do in your spare time?***

This is what I do in my spare time! I'm a differential geometer and work on low-dimensional gauge theory and topology, at least when I'm not teaching. I've spent the week interviewing would-be students: as I sit typing this at 1 a.m., I'm listening to Bach's "The Art of Fugue". I write a small amount of poetry, rather slowly, but publish little.

Q. ***In the Inform manual, you wrote "the author of a text adventure has to be schizophrenic in a way that the author of a novel does not." What are the most critical issues faced by the programmer in combining storytelling with game design?***

The single biggest is to stop the player from getting stuck and getting bored; always think like the player as well as the designer. This means keeping many trails open at once, inevitably requiring a fairly "parallel" plot. This plot should be discovered rather than announced, so show, don't tell. Finally, don't forget to throw in distractions and sub-plots. This makes an unconventional story: long on situations which have to be faced sooner or later, short on chains of consequence and changes of heart or attitude.

Q. ***From the player's perspective, finishing a good text adventure game can be a little sad in the same way that finishing a novel means leaving behind your favorite characters. What advice can you give for creating memorable game environments?***

Atmosphere is enormously important and requires fine crafting. Once a genre has been adopted, stick fairly closely to it. If you're setting a game during the Cuban Missile Crisis, look through a library: find out what people were wearing, what other issues were in the news, how houses were furnished, what cars were being driven. Especially include things which now seem foreign: say, a puzzle which involves dialing somebody's name on a phone whose dial has letters as well as numbers (long since phased out, in Britain anyway). And, as I said before, I like to employ a form of repetition, in which the same elements recur but in different and unexpected ways: rather than being discarded as soon as they are understood or passed over.

Q. ***What are the most valuable things you've learned about designing parsers for your games? To what lengths must a programmer go in anticipating the actions players will attempt?***

Extremities, I'm afraid. There is no substitute for extensive play-testing. About one-third of the code in a good game will be devoted to blind alleys and responses to wrong guesses by players. I like to code up some ideas of my play-testers as alternative solutions: if they make sense and don't unbalance the plot, why not? As I recall there are now six different ways to open the medicine bottle in "Curses".

Writing a really general parser is a major but different undertaking, by far the hardest points being sensitivity to context and resolution of ambiguity. I suppose my main lesson has been not to take shortcuts. Less flippantly, that it is better to write a parser which understands too much (i.e. understands some strangely worded requests which it ought to reject) than one which understands too little (because it has too narrow a grammar). ⊗

Zork-based MUDs: Coming to a Corner of Cyberspace Near You

People are social creatures, and there might come a time when you feel the need to enjoy the company of others when you gear up to play your favorite I-F adventure games. When that time comes, you might want to check out one of the Zork-based MUDs proliferating online.

The text-based world of MUDs are a natural for I-F fans. Named for multi-user dungeon, a MUD is a real-time online group effort that reflects much of the entertainment that many of us enjoy in solitary: characters acting to accomplish a pre-set mission or acquire treasure, player interaction that could be friendly or combatative, new regions where you can boldly go, map and explore.

Unlike playing an actual text adventure game—a static, pre-written, pre-coded endeavor—the experience of visiting a MUD is different every time because you're affected by what others are doing online. There are also various categories of MUDs with their own formats; a DikuMUD, for example, has a combat-oriented undertone and a strict hierarchical class system for its characters.

How can you find out what Zork-themed MUDs are up and running? A Usenet newsgroup, rec.games.mud.announce exists for just that purpose, offering a forum for new MUDs to broadcast their presence and lodge informational articles about themselves. There are also related newsgroups, such as rec.games.mud.lp and rec.games.mud.tiny, that are all about MUDs that have that particular format,

Below, we take a look at two current operating Zork-based MUDs:

- **ChicagoMUD**
  (telnet to jive.rahul.net 2600)

  Nino Ruffini operates ChicagoMUD, a DikuMUD derivative that is geographically based in the Zork empire, including such areas as the Barracks of the Antharian Guard, but also borrows freely from many other settings in mythology and history. One of the biggest problems faced in setting it up, according to Ruffini, was "adapting the puzzle-oriented Zork world into a MUD format that is basically hack-and-slash." On the plus side, however, using the DikuMUD format meant that little had to be built from scratch. Ruffini said he compensated by creating solid text descriptions of the world and the characters as a whole. Some 700 visitors have dropped by ChicagoMUD during a recent two-month period.

- **ZorkMUD**
  (telnet to lestat.shv.hb.se 7890)
  (temporarily at pine.cs.virginia.edu 7890)
  home page: http://lestat.shv.hb.se/~zorkmud/

  ZorkMUD has been operating on a small scale since the end of 1993, and will open on a larger scale sometime this year, says staffer Tim Hollebeek. "Anyone is welcome to stop by as long as they understand we're still in the building process." In this MUD, the Zork geography will serve as only a backbone for new puzzles built in the spirit of Zork. ZorkMUD is based on a LPmud driver, which means that most of the game must be implemented from scratch.

  A major issue has been how to accommodate the Zork-type puzzle-oriented games for multiple users. "Questions like 'How can someone start a puzzle if someone else has already finished it? Does it reset itself? If so, do the objects involved move back to where they start? What happens if someone is holding one?' crop up often, and have yet to be fully resolved," said Hollebeek.

  One last issue, according to Hollebeek, is that ZorkMUD is coded in LPC, which is often too complex for many members of the IF/role-playing/MUSH community with less programming background. ZorkMUD staffers are working on interfaces to help ease the transition.

—Stuart Beach

Ed. note: *If you have any information about noteworthy or newsworthy happenings in the interactive fiction community, please pass it on to us at xyzzynews@aol.com. We're always interested in releases of new versions of design tools, innovations in new games or other new software, and related Web sites or BBSes.*

10 Hints for Great Game Design

by **C.E. Forman**

You've solved every Infocom game ever released. You've FTP'd countless text adventure games from Internet sites in a desperate attempt to quench your insatiable thirst for interactive fiction, but still it's not enough. So you decide to take the final step, to write your own parser adventure. But—how do you know for sure that people will like it? How can you avoid making the same mistakes you've seen in many of the quests you've been playing for years? What exactly constitutes a "good" text adventure game?

That's what I'm here to help you with. I've taken it upon myself to analyze my favorite works of interactive fiction, determine why they're my faves, and compile a list of their common characteristics that first-time adventure writers can use for reference.

Keep in mind that this is not an article on programming a game. These ten tips deal exclusively with game design and the authoring of the game's storyline. My intent here is to point out the most common mistakes beginners make, and identify methods of avoiding falling into these traps.

You may be asking yourself, "Who is this guy anyway? I've never heard of him in my entire life!" True, true. I have been a fan of interactive fiction since I was seven years old, and I have even written a couple of my own adventures. That you've never seen my name anywhere should tell you that my games have fallen prey to most if not all of the problems I'm pointing out in this article. (Of course, I was confined to using BASIC as my sole programming language. Plus I was only twelve at the time.) Right now, I'm working on a couple of new games, and perhaps this time they'll turn out to be quite a bit better. Hopefully you'll be able to learn from my mistakes, too.

1. *Develop a good parser.*

This is the single most important element of any work of interactive fiction. Unfortunately, it's also the one most frequently neglected by beginners. Even the most cleverly designed adventure isn't going to hold players' interest for very long if they have trouble communicating what they want to do. The earliest adventure games, such as the original "Adventure in Colossal Cave" and the Scott Adams series, used crude, verb-and-noun parsers that accepted only two words in each command. Due to the limitations of computers in those days, a standard parser's vocabulary was often very limited, leaving gamers dissatisfied.

The Zork Implementation Parser introduced by Infocom in the late 1970's is really the accepted standard for parsers today. If you're using an IF design tool such as Inform or TADS, developing a good parser isn't as much of a problem. If, on the other hand, you've decided to write your own parser, pick yourself up a good thesaurus and use several common synonyms for each noun and verb. Make your puzzles the challenge of your adventure; don't force players to "guess the verb."

In addition, the more options you can supply your parser with, the better. An "undo" command, built-in hints, the ability to allow players to configure the function keys as typing shortcuts, and automatic mapping will all contribute to the reduction of frustration on the part of the player.

2. *Use good puzzle structuring.*

Don't just force players to wander aimlessly from one puzzle to the next, halting their progress completely until they solve the only available puzzle. Branch out your puzzle structure and make it as nonlinear as possible. Interweave your puzzles with one another and allow players multiple paths through the adventure. That is, don't make players solve the puzzles in the same order every time; give them some flexibility. The only point in the game where there should be only one path for the player to follow is at the conclusion, where all the branches of puzzles come together to form a final challenge.

Puzzle connectivity is also important. Make sure each puzzle "fits in" with all the others. If you have an extremely challenging puzzle, but you can't make it fit logically into your adventure, don't just throw it in for the sake of using it. Save it and use it in another game, where it *is* appropriate. One of the biggest abuses of this that I've seen comes in the form of mazes. Often adventure writers will simply throw in a maze to make the game more difficult, when in reality it is totally inappropriate and has nothing to do with the game at all. Making maps of games is tedious, and mazes are generally frowned upon in adventure games today, unless they have a truly unique twist (such as the catacombs in "Leather Goddesses of Phobos"), or can be solved without mapping (such as the wet tunnels in "The Lurking Horror"). In summary, designers should ask themselves this: "Is this puzzle connected to the game in some way, or is it in the game merely for the sake of its own existence?" If it's the latter, you should probably consider scrapping it.

3. *Balance the difficulty of your puzzles.*

What's the fun of getting all the way to the end of an adventure only to discover that the final challenge is the easiest puzzle in the history of the universe? In most cases, the best adventure games are the ones that curve the difficulty of their puzzles. Keep 'em fairly simple at first, to allow the player to get into the game, then gradually raise the challenge as players go deeper into it. Don't get me wrong.

It's perfectly okay to throw in a difficult or obscure puzzle or two in the early stages of the game, but the key is not to overwhelm the player at the outset.

Of course, the primary deciding factor as to the difficulty of the puzzles should be how difficult you've chosen to make the game as a whole. If you're writing for expert players, design your puzzles accordingly. If beginners are your target audience, include a lot of simple, one- or two- step puzzles. In all cases, after a particularly arduous puzzle, reward the player with a few simpler ones. You'd be surprised at how many players lose interest when a game's puzzles aren't balanced.

4. *Make your world and puzzles logical*

This one is basically just common sense, but it's still sometimes overlooked. If you're writing a sci-fi adventure, pay attention to the laws of physics. Don't let players enter the vacuum of space and survive without spacesuits. Realism is less of a problem in fantasy games, as much can be justified by the use of magic. The point, though, is to make sure everything makes sense in some way. This is especially important in the area of puzzles. Avoid making players do things that have no logic or purpose behind them. The more realistic your adventure, the more it will draw players in.

5. *Be descriptive.*

You've created a whole other world, so why not let the player enjoy the beauty of it? How many times have you played a game with such lame location descriptions as "You are in a forest," "You are at the bottom of a tall cliff," "You are outside a cave," etc.? The term "interactive fiction" is not an arbitrary one—players are essentially exploring a form of writing, much like a good novel, and adding their own input to it. So let the player see the world you've created, much like your favorite fiction authors let you see theirs.

Take care, though, not to overwhelm your players with prose. If you give them little opportunity to interact, they just might decide that they may as well be reading a book. Don't get too bogged down in descriptions. Usually half the screen is the absolute maximum for a room or object description, and this limit should only be reached on rare occasions. However, if a particular puzzle requires a lot of text in order for the player to see it, one or two full screens are acceptable. (A good example of this case is the mirror box in "Zork III".) Don't make players read the text over and over unless they want to, though. Make sure your parser has the option of changing the length of room descriptions. Using phrases such as "You are in the forest" the second time a player goes there is perfectly acceptable. (Just make sure that players can still get a better description if they want it.)

While we're at it, I'd like to mention one variation on this subject. Most players, when writing good room descriptions, like to include several objects or features in each location (for example, a tavern might have a fireplace, a bar, and several tables and stools). Nothing is more aggravating than typing "EXAMINE THE STOOLS" only to be told, "I don't know the word 'stools.'" This is guaranteed to instantaneously shatter the fantasy and destroy any hope of players ever really getting into the game. Do this enough, and you'll alienate them forever. If you're going to put an object in the location's description, you'd better let the player interact with it, even if it's only in a limited way. Just a message saying, "There's nothing special about the stools." will suffice.

Incidentally, I feel that this is one of the biggest problems with the Zork-based MUDs I've played. Players see that term, "Zork-based," and they telnet in expecting the same level of realism that Infocom gave us, and unfortunately, they rarely, if ever, get it. I myself have on occasion experienced difficulty in simply trying to interact with what the game claims is in the scene with me, and I'm afraid this is the rule rather than the exception.

6. *Be fair to the player.*

I know, I know. Life isn't fair. Never has been, never will be. But adventure games aren't real life; they're a form of entertainment. And the only way players will be entertained is if they're treated fairly. Here are some general guidelines you should follow to ensure that this is the case:

a. Don't let the game get into an unsolvable state too much, without giving the player some indication of it. There's no definite line here, so you'll have to use your own judgment. As an example, consider the KULCAD scroll in Infocom's "Enchanter". (Warning! Spoiler to follow…) You are supposed to use this spell to dispel the illusion of the infinite winding staircase, as this is the only way to overcome this particular obstacle. However, you can also use KULCAD to get rid of the guarded doorway and the Gordian Knot around the jeweled box. If you do one of these, though, the scroll is gone and you can't win the game. Despite the frustration that could be caused by this, I still don't consider it unfair, because if you use the spell on anything other than the stairs, your master Belboz will appear before you and warn you that the evil Krill has been alerted to your presence. You receive a definite hint that maybe there was a better way. On the other hand, I've heard numerous complaints about the game "Curses" because you can inadvertently do something out of sequence and blow any chance of being able to win, and no indication whatsoever is given. So save yourself and your players a lot of trouble, and give some kind of message if they unintentionally do something to get themselves stuck. (On the other hand, throwing all your possessions off a cliff is not a very smart move to begin with, and players should be able to figure this out without a hint. Only tell them they've screwed up big-time if there's a chance they can't determine that for themselves.) Incidentally, a good way to allow players to keep going after they've lost an item is to allow them to re-obtain the item in the place they originally got it. For instance, allow them to pick another apple off the tree if they eat or lose the one they originally got. However, when you're dealing with a unique item, such as the KULCAD scroll, this isn't a feasible option. Again, you'll have to use your own judgment here.

b. Don't force the player to have too much foresight. Inventory management is a crucial part of an adventure game in which the number of things a player can carry is limited. Often players will wander into a new location and only be able to take so much along with him. Obviously, if they're going into a dungeon they'll need a light source, a weapon, and probably some food and water, but if they're going to need something less obvious, you'd be wise to provide a hint beforehand. Of course, players can always restore, but going through a lot of moves to get back to where they was before can be frustrating, and too many save files can become difficult to keep track of. It's best to give players a general idea of which items they won't need and thus can leave behind. Don't make them pick and choose too much. A few very good games I've seen are designed so that the player's inventory is pretty much managed as the game progresses. That is, the player uses two items to solve a puzzle, thus removing them from his inventory. Then she finds another object and adds it, and later gets rid of it in another puzzle, and so on. If you have items that don't have multiple uses, this is a good technique to use.

c. Don't overwhelm players at the start. If you have a large area they can explore in the beginning, you'd be wise to point them in the right direction to start with. Legend Entertainment's "TimeQuest" is a perfect example of this. With 80 different time-places to visit, any of which can be reached from the beginning, it's vast enough to make the player feel burdened. But this game directs you to Rome in 44 B.C. from the start, which gives them a sense of direction and helps you establish a path through the game. In addition, the most crucial time-places are all listed in the game's documentation, so the player knows where the important places are. This makes it easier to get started.

d. Don't put off the entire reward until the end. Congratulate players when they solve a difficult puzzle, possibly by giving them a special item or power. According to Joseph Campbell's monomyth, the challenges a hero faces become more and more difficult as his quest continues, but the rewards become greater. This should apply to adventure games as well.

e. Don't create puzzles that absolutely have to be solved within a specific time frame, unless you give the player a reasonable hint.

f. Include good error messages in your program to tell players if they're doing something wrong, but don't insult players in the process. Be clever, but not verbally abusive.

g. Random events are good for spicing up adventure games, but never, EVER base the decision of whether a player lives or dies upon the outcome of a random-number generator. I mention this because I did it once. The player had to cross a pit by placing a wooden plank over it and then walking across. But you fell in ⅓ of the time anyway. Talk about frustrating! The only instance where this is at all acceptable is when there is an alternative solution that is not random. For example, in Sorcerer, players have a 10% chance of successfully jumping a gorge, but if they use a flying spell, they'll get across every time.

You might be saying, "Well, this is my game, so I'll do whatever the hell I want, and I don't care whether the player thinks it's fair or not!" Keep sending this attitude, and pretty soon you won't have any players who care about finishing your adventure. Book authors who don't show respect for their readers don't stay book authors for very long. The same holds true for interactive fiction writers. Players are your lifeblood; they keep your game alive. If you want to write games solely for your own pleasure, that's fine, but you won't gain any recognition from doing it. Treat your players as you would treat a paying customer, because after all, that's essentially what they are.

7. *Don't be afraid to kill a player off.*

At the same time, don't hold the player's hand all the way through the game. Let them experience a game death if their actions aren't clever enough. Dying is a natural part of adventure games. Can you picture what the Zork Trilogy would be like without the constant threat of being eaten by a grue in a dark place? And just try to imagine "The Lurking Horror" without death! Which would you rather see after doing something intentionally stupid in an adventure—a detailed, possibly amusing, account of your alter-ego's untimely demise? Or a lame message saying, "That would kill you, so I'm not even going to let you try it"? Believe it or not, it's FUN to try to find new and inventive ways to kill off your character (especially after you've already finished the game). Pampering players with feelings of invincibility is only going to make them severely disappointed. And besides, this is one of the few ways you can get away with murder nowadays. If you've put an UNDO command into your game, don't be afraid to let players use it.

8. *Concentrate equally on creating challenges and building the story.*

Puzzles are fun, but the story itself should be the main point of an adventure game. Rather than having the player wander aimlessly around solving puzzles, develop the story as the player moves along. An unexpected plot twist or the introduction of a new NPC can really liven things up, especially when it occurs in the midst of a good puzzle. Puzzles alone can only carry a game so far.

Another thing to keep in mind is that a game should have a good introduction and ending. Actually, an introduction is optional. Some writers may prefer to simply have the game begin as soon as it loads, much like “Zork I”, while others may choose to follow the route of “Beyond Zork” and have introductory text spanning several screens. A few games, such as “Zork Zero” and “Demon’s Tomb”, even have short prologues—opening sequences which play much like the game itself, but which exist for only a limited number of terms. Any of these methods will suffice.

A good ending, though, is indispensable. Is it worth struggling through a game just to be rewarded with the words, “Congratulations, you win”? A good game ending should tie up any and all loose ends the story may still have, pave the way for the sequel if you’re writing a series of games, and leave players feeling as though they have truly accomplished something. Good endings will be read again and again by players, but I can guarantee that a lame ending will only be seen once.

9. *Include good information sources in your game.*

Most of the time, when writing the game’s introduction, you’ll want to tell the players only so much about your world. Let them learn the various intricacies and details of it themselves. If your world is vast and complex, build several sources of information into your game to help the player accomplish this. You could implement an encyclopedia (Infocom uses these in several games), newspapers, a computer database, or some other form of information storage (the tape spools in Stationfall come to mind). In addition, you might want to make one or more characters act as primary information sources. The player could then ask those characters about various people, places, or things in the game. The more you tell the player about your world, the more complex and realistic it will appear.

10. *Keep the game’s longevity intact.*

A lot of good adventure games become dust collectors after players have solved them. Often this can be prevented, or at least delayed, by a little extra effort on the part of the author. Give some of your puzzles multiple solutions. Think up imaginative ways of dying and humorous tricks for the player to try. Some adventures even have multiple endings depending on various things the player has done (or not done) during the course of the game. All of these things can keep players interested for quite some time after they’ve been through the entire game. The “Zork” and “Enchanter” series are particularly good examples of this tip. I’m still finding things buried in them that I never knew existed. A little extra programming can go a long way.

And there you have it. If you see yourself making any of these mistakes, you might consider rethinking at least part of your original design. Designing a good text adventure is a long, complicated, involving, and extremely frustrating task, but in the end it can be very rewarding. Good interactive fiction is always in demand, and I sincerely hope I have enlightened you and guided you toward successfully creating your own.

Busted

(release 4)

Parser: Inform
Authors: Jon Drukman (jsd@cyborganic.com), Derek Pizzutto, and Mike Wertheim for Scumbag Software
Availability: ftp.gmd.de/if-archive/games/infocom; CompuServe
Supports: Infocom ports

When you're ready to tune in, turn on, and drop out to play computer games, "Busted" from Scumbag Software is a very appropriate choice. Written in Inform, this release is an updated port of an earlier ADVSYS version by Jon Drukman. "Busted" is set in a collegiate environment where police crackdowns on drug possession are rampant. The game begins as you return to your dorm room and listen to a disturbing answering machine message. The police have just busted your friend Keith and will be on the lookout for you next. In order to save your butt, you need to track down and get rid of Keith's address book, plus any incriminating evidence that you may have left around campus. Along the way, you'll need to solve a series of highly aggravating but rather original puzzles in order to progress to other parts of the game.

As in many other text adventures, "Busted" forces you to fulfill some basic human needs—in this case, eating and sleeping—before you can do much else. Beginners are likely to get fed up very quickly. But if you get past this point of the game, you'll probably have invested so much time and effort into solving these two tedious little problems that you'll feel compelled to play it through to the end! After those initial troubles, you must still avoid lingering in dangerous locations or having a bad trip.

A little prior knowledge of popular recreational drugs and their effects will take you far in playing "Busted." As I played, my ignorance in this area proved a liability (especially embarrassing because the college I went to had such a pervasive druggie atmosphere). But there are some fun references to popular culture, such as the Cheech 'n' Chong lunch box and the allusions to "Tommy" in the pinball arcade.

Many of the puzzles hinge on interacting with or acquiring objects from the numerous NPCs that populate the game. You'll run across other students, workers, mindless bureaucrats, and others in positions of authority. There's also a mysterious stranger roaming the campus, injecting people with a hallucinogenic substance, and even the police are puzzled about his motives. As might be expected, you can't really talk extensively with any of the NPCs, and their characters run pretty true to stereotype. Some, such as the nose-picking Dining Commons line lady, have amusing traits. You'll need to either possess or conceal certain items before accessing several rooms in the game. Most of the locations reflect ordinary college locales, such as a dismal cafeteria, dreary campus center, and an antiseptic heath center. Unfortunately, too few of the items mentioned in the room descriptions could be accessed or described at any length.

There are no online hints, which was exasperating when trying to guess the proper syntax for executing an action or the proper order for making a series of moves. In the library, for example, there's no indication that you must put your ID in the computer slot before typing the code for the book you want. I typed first, then put in the ID; when that didn't work, I spent the next hour thinking I must've had the wrong code. If you type HELP, the response begins, "This game is pretty damned easy if you ask me," but does go on to list how to contact the game's author for hints.

By far, the most frustrating aspect of playing "Busted" centers on the wildly uneven quality of the parser. For example, when the line lady says "Let's see some ID," why does SHOW ID TO LADY work but GIVE ID TO LADY produces "The Dining Commons line lady doesn't seem interested"? Why does LOOK IN ASHTRAY produce nothing, when LOOK AT ASHTRAY or EXAMINE ASHTRAY does? Complex sentences had to be abandoned entirely. When my efforts failed, I could never be sure if it was because I had the wrong solution in mind or if it was because I hadn't yet hit upon the single acceptable way to phrase that action.

Somehow, though, it was enormously satisfying to solve each logistical problem and make even the slightest headway. Despite some initial turbulence, the game is a lot of fun when you get immersed in it. If you can tolerate the limited parser and subject matter, you'll find the game fairly enjoyable. Although Nancy Reagan would urge you to just say no, "Busted" is an entertaining way to enjoy the vicarious thrills of outsmarting the vice squad.

— Eileen Mullin

CosmoServe

Parser: Adventure Game Toolkit
Author: Judith Pintar
(76636.2067@compuserve.com)
Availability: CompuServe
Supports: AGT ports (PC, some early Macs, Atari ST)

If you spend your days compulsively logging on and offline, or if you regularly find yourself at the mercy of your computer for completing an important project by a fast-approaching deadline, you'll appreciate "CosmoServe: An Adventure Game for the BBS-Enslaved." An AGT game written by Judith Pintar, "CosmoServe" combines a compelling story of the perils of a freelance computer programmer with a wry parody of the chat culture found on CompuServe and other online services.

As R.J. Wright, a programmer who also fixes indoor plumbing to make ends meet, you have several problems you must solve—and quickly. The major task at hand is completing the code for a program you must deliver to an important client first thing tomorrow morning. You're so seriously in debt (all those CosmoServe charges!) that it may be the end of your business if you let down this client. The clock is ticking, but the code won't compile. Is it a hardware problem or is it the Pascal? And now you've forgotten your new CosmoServe password; how are you going to check your email to see if the technical support sysop has replied to your plea for help? You must also contend with mysterious computer viruses, gaining access to virtual reality mode, fraudulent charges on your CosmoServe account, and a dangerous online stalker.

Almost all of the game's 86 locations are online on the CosmoServe service or otherwise in the plane of virtual reality; you can also putter around R.J.'s house or the C: drive on your computer. Examining all the files in all directories on your hard drive is crucial to finding clues, as is entering each area on CosmoServe. The faux DOS interface was as much fun as the inside jokes about CompuServe. I especially liked the replication of CompuServe's electronic mall and being accused of "lurking" on an online conference. You must also keep an eye on the ticking clock so that you can be on time to rendezvous with online contacts at scheduled conferences. The game has some limited sound capabilities, which consist of differently toned beeps for logging on, when a virus takes over your computer, etc.

The game's scoring system is vast (up to 1,000 points). Points are awarded generously, even for such actions as reading useful files or messages. The parser does not return a message when points are awarded, so players should keep checking their points in the status line or they may not even realize when or why they've been rewarded.

In "CosmoServe," getting ahead hinges upon your ability to absorb information and put it to use. While the initial puzzles are fairly easy, the online portion of gameplay is complicated by malevolent forces such as the online stalker and the impending time deadline. Limited hints are available, nicely incorporated into the theme of online services. When you're logged on to CosmoServe you can type GO HINTS and leave a message for the hints sysop, choosing from an menu of available hints. But naturally, it's frustrating when the question you need help with isn't listed on the menu. Help is also available through a nosy NPC named Aunt Edna, who appears if you have trouble with initial game play and assists with the various steps you must take to find your lost password.

Although the game's hints are charitable, they are at best only mild spoilers. And there are still many, many ways to get into trouble. It's extremely easy, for example, for your computer to be destroyed by a virus, or for you to reach your credit limit due to too much shopping or fraudulent charges to your account. You might also find yourself the target of a police sting operation, or so overwhelmed by the VR experience that you conk out for the rest of the night.

You can save your game at any point, and it's very helpful to do so, in case of unexpected crises, such as an unauthorized intruder who changes your password and locks you out of using your own account. It can also take a number of false starts or second-guessing your better instincts to decide who you can trust online and whose posts you should respond to.

"CosmoServe" was a first place winner in the annual AGT game writing contest for 1992. The game is freeware, but you can register your game by sending the game's author your name, address, and the meaning of life in 20 words or less. It's an excellent intermediate game with broad appeal for all hackers, text adventurers and BBS addicts.

—Melissa Katz

Curses
(version 12)

Parser: Inform
Author: Graham Nelson (nelson@vax.oxford.ac.uk)
Availability: ftp.gmd.de/if-archive/games;
CompuServe; America Online
Requires: ZIP interpreter

The family's going on a vacation trip to Paris, France, and everyone's busy packing for the trip. Soon you'll be seeing historical monuments up close and personal, wandering through beautiful gardens, taking pictures for a slide show back home… But wait—there's that tourist map from your trip five years ago. You could save a few francs by digging that old thing up, and you could use a bit of a break from packing. It should only take a few minutes to find the dratted thing in the attic…

That's what you think.

As the story of "Curses" unfolds, you'll indeed find historical monuments (good as new), beautiful gardens (your own), and slide shows (as you've never before seen them). You'll also find numerous puzzles ranging from simple (not many of those) to difficult (lots) and hair-tearingly frustrating, spread across the years from ancient Egypt to the present. And you will find literary allusions, references to historical events and legends, and a family history that literally meanders all over the map, but remains centered about the ancestral Meldrew Hall in England...

"Curses" is rooted in the traditions of Infocom games where you must wander about your house and the various scenes you'll be able to unravel, pick up objects and use them to aid your quest, answer riddles, and above all else, examine everything you see so that you won't miss vital clues. As with other works of interactive fiction, you'll be able to save the game at any point, which will come in handy…You aren't likely to die very often in this game, but some scenes can only be entered once: this means that if you miss something important and realize it later, you'll want to go back to before that scene.

"Curses" well qualifies for the title of "interactive fiction" as compared to the "adventure game" which strings together puzzles with little relation to the storyline. Apart from the descriptions, which are generally well written with an eye to atmospherics, you'll find bits and pieces of family lore that weave back and forth across a historical canvas, sewn together by your efforts. When (if) you finish the game, it should all make sense… more or less.

Most puzzles depend on the traditional verbs: opening and closing things, unlocking things with other things, pushing, pulling, moving things, inserting things in other things, or turning things on or off. There are a few NPCs, such as Austin, Jemima's cat, but don't expect to be able to converse intelligently with them—you can give them things, occasionally—manipulate them in other ways, or ask about various things, but they aren't much for talk.

The parser gets most of the things that Infocom games will, but there are some points at which you must use correct phrasing — if you think that something OUGHT to work a certain way, try phrasing it differently or using a different verb.

Mercifully, "Curses" manages to be original in its puzzles, which will depend in large part on the history that you can uncover with the help of a few references (some of which are even included in the game). Clues to these puzzles can come from surprising sources, that appear at first blush to be unrelated. (when I say you must examine everything, I'm not kidding) The game's on-line hint system has been personified in the form of supernatural agencies—alas, the hints present are rather cryptic, and if you can't figure out a puzzle from the hint, there won't be any more hints. These agencies also may not recognize some puzzles you want to ask about, which can be frustrating.

"Curses" release 12 expands its map and puzzles in many ways over previous versions, which makes this game even tougher and more frustrating. If you can beat "Curses" single-handedly on this version, then you may accredit yourself a most accomplished gamer. If not… don't feel bad, almost everyone else had to ask for help. There are 550 points, and the score system is annotated — if you ask for a 'full' score, you'll be told what points you got from what source. Some of the annotations can be rather amusing.

If you liked the Zork trilogy, and you have a fondness for mythology and obscure things, then you will probably enjoy "Curses". But beware! This is no game where you can sit back and watch the daisies grow—keep your mind sharp, and above all…examine everything.

—Conrad Wong

Veritas

(release 1.2)

Type: TADS
Author: Jim Reese (jreese@leland.stanford.edu)
Availability: Released as of 1/1/95. Contact author for more availability information.
Supports: TADS ports

"Veritas" — which, this game tells us, is Latin for "truth," as well as being the Harvard motto — is set at, where else, Harvard University, as you attempt to meet the standards for graduation. And with standards like these, no wonder I didn't go to an Ivy League school; to graduate (and, not so coincidentally, win the game) you must present your senior tutor with a list of items that ranges from the expected (a senior thesis) to the bizarre (an edition of the Gutenberg Bible), all of which can be obtained through your travels in, around, and under the Harvard campus.

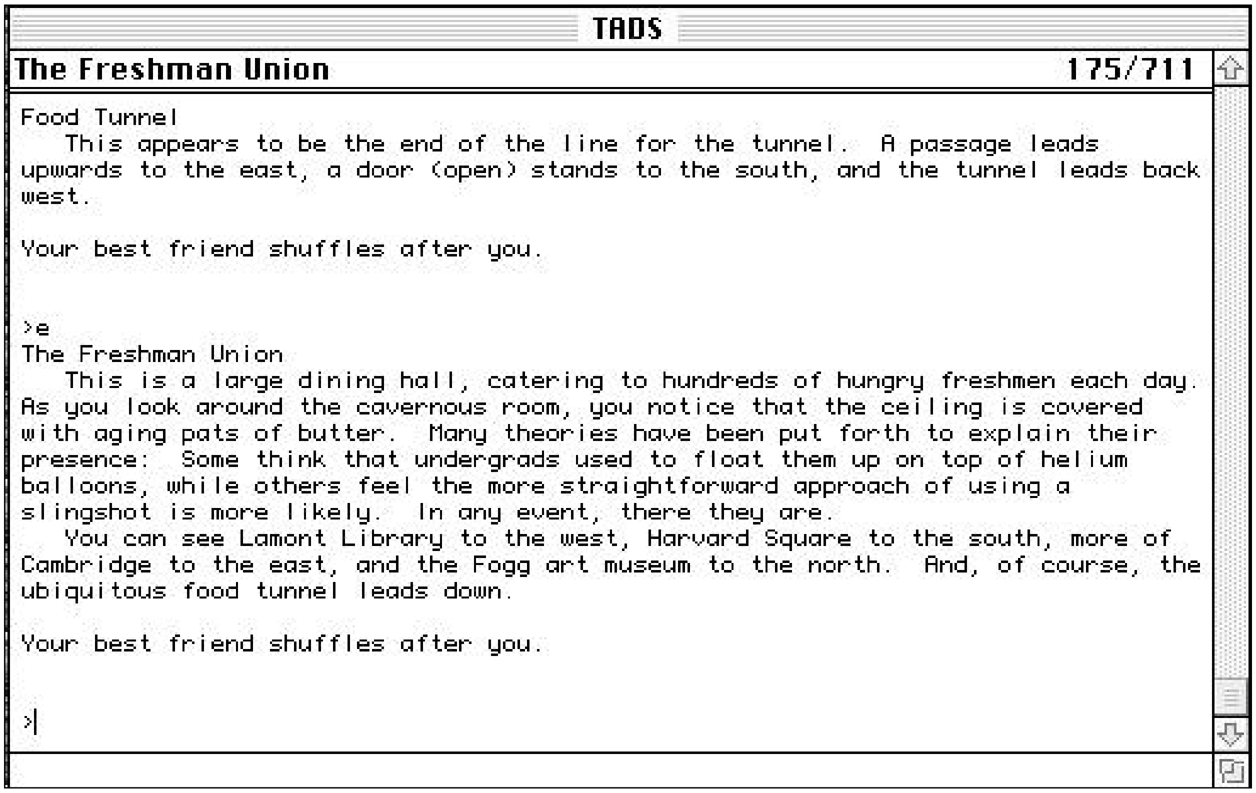
The puzzles you must solve to complete this scavenger hunt range from the mundane to the harder-than-average—a couple I wouldn't classify as puzzles at all, since you can easily solve them without even trying. (Others, such as one involving creative uses for a yardstick, are quite clever.) If I have one complaint about this game, it's that there's a bit too much brute searching involved; it seemed like one too many times that I knew just what I needed, but wasn't sure what object I still needed to look behind, or under, or inside to find it.

But this is nit-picking, because the true genius of "Veritas" isn't in the puzzles, but in the details. Writer/programmer James Reese has worked up an incredibly detailed virtual Harvard, with intricate descriptions of everything from the tapestries in the Harvard Lampoon castle to the works of art in the Fogg Art Museum (or, after you've swiped them, wry commentary on the sad times we live in that have left nothing but blank spaces on the walls).

If you hate red herrings, be forewarned: probably one-third of the items in "Veritas" are there for nothing but atmosphere, but they do a terrific job of providing that — I feel like I came away from the game with a decent understanding of Harvard culture, and even Cambridge geography. I can't wait to meet a Harvard grad now, so I can ask them if there's a statue of John Harvard, "Harvard founder," that depicts another man altogether, the year the college was founded, and even that he *was* the founder of the college…

"Veritas" is a fun play, and well worth the $10 shareware fee which should help encourage Reese, now in med school at Stanford, to finish work on the promised sequel, "Club Med."

—*Neil deMause*



```
TADS
The Freshman Union                                              175/711
Food Tunnel
   This appears to be the end of the line for the tunnel.  A passage leads
upwards to the east, a door (open) stands to the south, and the tunnel leads back
west.

Your best friend shuffles after you.


>e
The Freshman Union
   This is a large dining hall, catering to hundreds of hungry freshmen each day.
As you look around the cavernous room, you notice that the ceiling is covered
with aging pats of butter.  Many theories have been put forth to explain their
presence:  Some think that undergrads used to float them up on top of helium
balloons, while others feel the more straightforward approach of using a
slingshot is more likely.  In any event, there they are.
   You can see Lamont Library to the west, Harvard Square to the south, more of
Cambridge to the east, and the Fogg art museum to the north.  And, of course, the
ubiquitous food tunnel leads down.

Your best friend shuffles after you.


>
```

Odieus

Parser: Inform; also available in AGT
Author: Inform port by Teo Kwang Liak (bh71139770@ntuvax.ac.sg); AGT version by David Malmberg; original author unknown
Availability: ftp.gmd.de/if-archive/games; AGT version on CompuServe GAMERS forum
Supports: ZIP interpreter

"Odieus" or "Odieus' Quest for the Magic Flingshot" is a much-recycled game, The version currently residing on ftp.gmd.de/if-archives/games was ported to Inform by Teo Kwang Liak. The game (still a beta) was written as an Inform programming exercise; it's a straightforward translation of the AGT version written by David Malmberg some years back, which in turn is an adaptation of a LADS adventure by an unknown author circa 1988. Whew!

So, what's the appeal? Why has this game endured for at least seven years? I wanted to see if Odieus was interesting as a game, not just a coding exercise. I also played both the AGT version and the Inform version to see if there were any noticeable differences in play or in difficulty.

It's easy to see why "Odieus" is a good choice for experiments in programming. It's a very short game (24 locations) with uncomplicated puzzles and every object in the game (except for a red herring, literally) has a single use. As the game's introduction tells you, your name is Odieus and you come from a long line of sorcerers and enchantresses. You're well-educated in the art of spell flinging, but unfortunately your magic flingshot has been stolen by your bitter archenemy Blackwing. Your mission consists of gaining entrance to Blackwing's lair and discovering where he's stashed your prized possession. You have to get past several guarded locations (staffed by a gorilla, a pride of lions, and a sleeping giant, among others). Solutions tend to involve the arbitrary use of objects, rather than logical ones, especially for the final puzzle in the game.

For avid players looking for a new challenge, though, there's little about "Odieus" that's attention-getting. The story is thin, character development is nil, and the puzzles aren't terribly intriguing. In general, magic and sorcery themes leave me cold. But even if you are an fan of "Enchanter"-type games, be forewarned that the sorcery is just for atmosphere and there aren't any scrolls to learn or spells to cast.

The best audience for "Odieus" is probably beginning programmers, which explains its many reincarnations. Every language has its own conventions for ensuring that conditions have been met (i.e., objects are present, objects are used correctly), so I'm sure it is a good exercise to translate what must be a fairly simple program (speaking as someone who knows nothing about programming) into your language of choice.

I did find the Inform version kind of buggy. My first complaint is that the parser can be contradictory. If you try to chop down a tree with an axe in one room you get the stock response to chopping anything: "Cutting things up in this game is never useful," but a tree does come crashing down one room over when you chop at it. A new default answer is definitely needed. Also, most of the room descriptions failed to list all the directions in which you can go.

My biggest problem in playing the Inform port, though, was trying to find the right syntax for solving one problem: cooling off the hot springs. I could do it with no problem in the AGT version, but I couldn't get the parser in the Inform version to accept anything I came up with. So, while I managed to complete the game in the AGT version, I remained stuck at this puzzle (about one-quarter of the way through the game) in the Inform version. The bug is probably something that would be easy to fix.

I didn't find these specific problems in the AGT version, but overall the parser in the AGT version was much more frustrating. The vocabulary was very limited and the parser gave few clues if the names you tried to use for objects were inappropriate.

In both version of "Odieus" you wind up with 150 points, and you need to get all the points to finish the game. (The AGT version, though, will award you an extra 10 points if you print the order form for buying your own copy of the Adventure Game Toolkit!) While the story of "Odieus" won't keep anyone on the edge of their seat, I'm sure it would be interesting to compare the coding challenges from a programmer's perspective. I'm sure the TADS and ALAN versions won't be far behind…

— Lauren Meckler

Special Delivery: 5,000 Mailboxes in TADS

by **Neil deMause**

Always, it came back to the mailboxes.

I am not a computer programmer. Before I decided to learn TADS and finally program the text adventure game I had been mulling over for years (“MacWesleyan”, due out any day now), my last programming experience had been in BASIC, on my Tandy Color Computer (with 16K RAM upgrade!) that disappeared into my parents’ closet somewhere in the early ’80s.

Still, I was doing okay—TADS is a very logical system, and helpfully spits back yards of error messages at you to let you know what you’re doing wrong—until I got to the mailboxes.

Here’s the scenario: A post office in a campus student center. In the post office, a wall covered in mailboxes (5,000 of them, to be exact). Only one of them is yours; the rest are just there are window dressing, and to thoroughly befuddle you until you discover how to find out your mailbox number. All I needed, then, so far as the programming was concerned, was to make two mailboxes, one yours and one not (which would stand in for all the other mailboxes)—or maybe even just one mailbox, which wouldn’t let you open it unless you gave it the right number. But how to make an object that would respond to any sentence of the form “VERB mailbox NUMBER”; this was my dilemma.

At the suggestion of the editor of this fine journal, I took my dilemma to the denizens of the rec.arts.int-fiction newsgroup. Surely, these programming experts would have come up with code for something as simple as a bunch of mailboxes?

My first response, from a well-known TADS programmer, mounted to a list of ways *not* to do it: make “mailbox” an article (which he called “pretty evil”), make “mailbox” an adjective of the object NumObj (“this seems dangerous”). Ultimately, he said I should try to use TADS’ Preparse function, a fabulously powerful tool that is as incomprehensible to me as Linear B.

My next message came in the form of an e-mail missive from someone in Finland whose name appeared to be entirely made up of random numbers and letters. (I’ve heard that this is common among European Net accounts. It must be just one of those Continental things that we Americans can’t understand, like national health care.) Though this person was undoubtedly well-meaning, his (her?) suggestions were complicated by the fact that his English syntax was about as good as my TADS code—a nice try, but ultimately just leaving me scratching my head and blurting out error messages.

Finally, to my rescue rode none other than MJR—Michael J. Roberts, the always-helpful creator of TADS. The obvious solution, he said, was to add an entirely new class of object, numberedObject, that would do what I wanted. Unfortunately, that would require rewriting the parser to handle the new syntax; he'd get back to me when it was ready.

Meanwhile, I had hungry beta-testers who couldn't care less about the nuances of TADS code—they wanted their mailboxes, and they wanted them now! I tried various workarounds, including the suggestions that the famous game designer had said were bad ideas. (He was right. They were.) Finally, I simply diverted all action from NumObj while in the Post Office to the mailboxes: now if you typed "OPEN 2000," it would try to open the mailbox with a "value" of 2000, and would only succeed if it was your mailbox, and so openable. For those players who insisted on typing a more sensible sentence like "OPEN MAILBOX 2000," I programmed in a special error message telling them how to phrase their request so that my bad programming could make sense of it.

This worked—most of the time. About one time out of ten TADS refused to recognize even properly phrased sentences. I began warning my beta-testers that the mailboxes were a "work in progress."

Another week, and another message from MJR. The new parser was ready, and new code for numberedObject. This employed a trick: you could have only one object (in my case, one mailbox), and TADS would make a *copy* of the object, with a value set to the number that was input. This way, a minimum amount of coding could result in a different response depending on what mailbox you specified. This could be it! I got it downloaded, changed the coding...

And it crashed. Over and over again.

Back to MJR for *another* new version of the parser. Fingers crossed, I ran the compiler, and started up my game. Cautiously I entered the post office.

```
> UNLOCK MAILBOX 2000
```

It unlocked.

```
>OPEN MAILBOX 2000
```

It opened. There, inside, was the postcard I had placed there! My long quest had ended! I happily open and shut the mailbox, removed the postcard and put in back inside.

Where, as Ensign Chekov would say, "it wanished." Gone, without a trace. The subroutine that deleted the duplicate mailbox at the end of the command sequence had efficiently taken my postcard with it, and there was no getting it back.

As *XYZZYnews* went to press, I was still waiting for an answer from MJR on this latest problem. I'm sure he'll have one—he's a smart guy. Meanwhile, my mailboxes sit there, happily gobbling up anything a hapless adventurer chooses to insert into their gaping maws.

Sometimes I wish my parents had never bought me that Tandy in the first place.

Ed. note: *I'm happy to report that MJR came through in the clutch, the mailboxes now work as planned, and Neil now has only fond memories of his Tandy.*

Q. I need to know how to get past the half-formed urchins (that are attached by their heads to a slimy thing below) in the caves below the campus in "Lurking Horror". After steam-blasting the rats, I go down there and they won't let me pass! Plus, every time I try and get the mummified hand from the peach tree, no matter what I do the flier-creature eats it! I can't get away from the roof with it! So: how can I keep the mummified hand?

A. To answer your questions in reverse order: have you logged onto the whiz-bang PC in the Terminal Room, where the game begins? If so, you should have the stone that you find in the dream sequence you enter while logged onto the PC. The stone is what you can use to scare off the dark flier outside the skyscraper roof. Just make sure you pick it up again after you use it there…

Your second question is more complicated to answer without giving major, major spoilers (despite the title of this column, we don't want to solve *everything* for you. We just want to give you enough info to get past one little holdup so that you can move on to bigger and better challenges). You're going to need an implement for cutting the wires; the urchin might be able to help you, if you can have the right effect on him. What you'll realize, though, is that you'll need a workaround for getting the cutter down to the basement. These games sure have their ups and downs, don't they…if you still need help at this point, we'll look for you next issue, same bat column, same bat game!

Q. I'm stuck annoyingly early in "Monkey Island 2: LeChuck's Revenge". To make the voodoo doll of Largo, I've gotten his hair, his spit, and his ancestor's bone, but I still need to get an item of his clothing. How do I get the ticket to get the clean laundry?

A. First, you need to get the bucket from near the laundry. Leave that area and go to the Swamp. Click the bucket on the swamp to fill it with mud. Return to Woodtick and go back to Largo's hotel room. Close the door (note that it won't stay completely shut) Balance the bucket of mud on top of the door, and wait for Largo to return (as he always will when you're in his room). As you might expect, when he comes in the bucket is knocked down and Largo's clothes are ruined. Follow him back to the laundry and eavesdrop on his conversation with the laundryman. Return to Largo's room and look behind the door for the laundry ticket. You can then go and use the ticket to pick up the clean laundry.

Q. In "Return to Zork", how do I get milk from the cow, or can I? And what do I do with the witch?

A. You'll notice that the reason you're having trouble milking the cow is because your hands are too cold. You need to create a fire and then warm your hands over it. I think you may need to pick up the straw (also in the barn) and then drop it. Light a match and then set the patch of straw on fire. Warm your hands over the fire, then milk the cow. Make sure you have carrots to feed to the cow afterwards.

As far as Witch Itah is concerned, she's miffed over a miscommunication with her former beau, Ben Fyshin (the guy you rent the boat from). Before you head over on the boat, show a photo of any female character to Ben; he'll start talking about his relationship with Witch Itah and ask you to give her a letter. When you deliver the letter to the witch, she'll let you take some items from her cave that'll prove very useful to you.

Q. In "Cutthroats", I can never shake McGinty and am always killed before the diving expedition even gets underway. How do you get to Point Lookout and the other meetings without him following you wherever you go? McGinty keeps following me, then Johnny gets really mad at me and I'm killed.

A. The key is avoiding McGinty when you're carrying your bank passbook. If McGinty sees you while you are carrying it, he will follow you and spoil everything, so you must absolutely avoid him when it's in your inventory. After you get the money from the bank, you can lock your passbook back in your room.

WHAT'S ON THE DISK…WHAT'S ON THE DISK…WHAT'S ON THE DISK…WHAT'S ON THE DISK…

The companion disk for XYZZYnews *#1 contains the following game files. It's a good deal for people who have slower modems—at 2400 bps, it'd take about three hours to download the contents of the companion disk. It's also a good deal for people with limited or no access to FTP sites or online services as a source for new games. If you're reading an electronic version of this issue, you can obtain this games disk with a print copy of* XYZZYnews *#1 by enclosing $3.50 for postage and handling with the coupon on the bottom of this page. If you play and enjoy these games, please pay the shareware fees as applicable.*

CURSES — An attic-wide search for a tourist map of Paris leads to an incredible journey to 1920s France, ancient Alexandria, sixth century Rome, and other locales. (See review this issue.) Freeware.

HUMBUG — You are sent to Grandad's for the holidays, but something strange is afoot. Why is there a time machine in the cellar? What would you do with a trombone, a terrapin, and half a pound of lard? Shareware, US$20 or UK£$20 *(PC disk only)*

ENHANCED — Here you are, trying to survive yet another day in the big city with no money, no job and no hope. As you walk down the street in the government research project area, you suddenly see a sign reading: "Volunteers needed for military research project! Do you want to make money? Do you want to help your country? Can you start at once? No special skills needed. Enter here." Everything on the list sounds good to you, so you enter the door and sign up. That's when your nightmare begins… Shareware, US$10.

VERITAS — Before you can graduate from Harvard in this game, you must complete a scavenger hunt that takes you through a labyrinth of food tunnels, the Harvard Lampoon headquarters, library stacks, and other noteworthy spots on campus. (See review this issue.) Shareware, US$10.

UNNKULIAN 1 — In "Unnkulian Underworld: The Unknown Unventure,"you are a slave and apprentice to the ancient wise man Kuulest. After his death, you are the only one who can to retrieve the Orb of Studosity and save the world from the reign of the evil Unnkulians. The first game in the engaging Unnkulian series. Shareware, US$10.

XYZZYnews **Magazine / Disk Order Form**

Checks and money orders preferred. Please send coupon with payment to: Eileen Mullin, XYZZYnews*, 320 W. 84th Street, Ste. 5B, New York, NY 10024.*

☐ Please send me a copy of the print version and companion games disk for *XYZZYnews* Issue #1. I have enclosed $3.50 for postage & handling. (Check one: Mac disk ___ or PC disk ___)

☐ I need just the companion games disk for *XYZZYnews* Issue #1. I have enclosed $2.50 for postage & handling. (Check one: Mac disk ___ or PC disk ___)

☐ Sign me up for a 6-issue subscription. I have enclosed $21 for postage and handling. (Check one: Mac disk ___ or PC disk ___)

☐ Please send me a copy of the upcoming *XYZZYnews* Issue #2. I have enclosed $3.50 for postage & handling. (Check one: Mac disk ___ or PC disk ___)

Full Name (please print): ______________________________

Street Address: ______________________________

City: ______________ State: __________ Zip/Postal Code: __________

Country: ______________ Email Address: ______________