A Guide to the Mazes of Menace:
Guidebook for *NetHack*

Eric S. Raymond
(Extensively edited and expanded for 3.4)

August 14, 2003

1 Introduction

Recently, you have begun to find yourself unfulfilled and distant in your daily occupation. Strange dreams of prospecting, stealing, crusading, and combat have haunted you in your sleep for many months, but you aren't sure of the reason. You wonder whether you have in fact been having those dreams all your life, and somehow managed to forget about them until now. Some nights you awaken suddenly and cry out, terrified at the vivid recollection of the strange and powerful creatures that seem to be lurking behind every corner of the dungeon in your dream. Could these details haunting your dreams be real? As each night passes, you feel the desire to enter the mysterious caverns near the ruins grow stronger. Each morning, however, you quickly put the idea out of your head as you recall the tales of those who entered the caverns before you and did not return. Eventually you can resist the yearning to seek out the fantastic place in your dreams no longer. After all, when other adventurers came back this way after spending time in the caverns, they usually seemed better off than when they passed through the first time. And who was to say that all of those who did not return had not just kept going?

Asking around, you hear about a bauble, called the Amulet of Yendor by some, which, if you can find it, will bring you great wealth. One legend you were told even mentioned that the one who finds the amulet will be granted immortality by the gods. The amulet is rumored to be somewhere beyond the Valley of Gehennom, deep within the Mazes of Menace. Upon hearing the legends, you immediately realize that there is some profound and undiscovered reason that you are to descend into the caverns and seek out that amulet of which they spoke. Even if the rumors of the amulet's powers are untrue, you decide that you should at least be able to sell the tales of your adventures to the local minstrels for a tidy sum, especially if you encounter any of the terrifying and magical creatures of your dreams along the way. You spend one last night fortifying yourself at the local inn, becoming more and more depressed as you watch the odds of your success being posted on the inn's walls getting lower and lower.

In the morning you awake, collect your belongings, and set off for the dungeon. After several days of uneventful travel, you see the ancient ruins that mark the entrance to the Mazes of Menace. It is late at night, so you make camp at the entrance and spend the night sleeping under the open skies. In the morning, you gather your gear, eat what may be your last meal outside, and enter the dungeon...

2 What is going on here?

You have just begun a game of *NetHack*. Your goal is to grab as much treasure as you can, retrieve the Amulet of Yendor, and escape the Mazes of Menace alive.

Your abilities and strengths for dealing with the hazards of adventure will vary with your background and training:

Archeologists understand dungeons pretty well; this enables them to move quickly and sneak up on the local nasties. They start equipped with the tools for a proper scientific expedition.

Barbarians are warriors out of the hinterland, hardened to battle. They begin their quests with naught but uncommon strength, a trusty hauberk, and a great two-handed sword.

Cavemen and **Cavewomen** start with exceptional strength, but unfortunately, neolithic weapons.

Healers are wise in medicine and apothecary. They know the herbs and simples that can restore vitality, ease pain, anesthetize, and neutralize poisons; and with their instruments, they can divine a being's state of health or sickness. Their medical practice earns them quite reasonable amounts of money, with which they enter the dungeon.

Knights are distinguished from the common skirmisher by their devotion to the ideals of chivalry and by the surpassing excellence of their armor.

Monks are ascetics, who by rigorous practice of physical and mental disciplines have become capable of fighting as effectively without weapons as with. They wear no armor but make up for it with increased mobility.

Priests and **Priestesses** are clerics militant, crusaders advancing the cause of righteousness with arms, armor, and arts thaumaturgic. Their ability to commune with deities via prayer occasionally extricates them from peril, but can also put them in it.

Rangers are most at home in the woods, and some say slightly out of place in a dungeon. They are, however, experts in archery as well as tracking and stealthy movement.

Rogues are agile and stealthy thieves, with knowledge of locks, traps, and poisons. Their advantage lies in surprise, which they employ to great advantage.

Samurai are the elite warriors of feudal Nippon. They are lightly armored and quick, and wear the *dai-sho*, two swords of the deadliest keenness.

Tourists start out with lots of gold (suitable for shopping with), a credit card, lots of food, some maps, and an expensive camera. Most monsters don't like being photographed.

Valkyries are hardy warrior women. Their upbringing in the harsh Northlands makes them strong, inures them to extremes of cold, and instills in them stealth and cunning.

Wizards start out with a knowledge of magic, a selection of magical items, and a particular affinity for dweomercraft. Although seemingly weak and easy to overcome at first sight, an experienced Wizard is a deadly foe.

You may also choose the race of your character:

Dwarves are smaller than humans or elves, but are stocky and solid individuals. Dwarves' most notable trait is their great expertise in mining and metalwork. Dwarvish armor is said to be second in quality not even to the mithril armor of the Elves.

Elves are agile, quick, and perceptive; very little of what goes on will escape an Elf. The quality of Elven craftsmanship often gives them an advantage in arms and armor.

Gnomes are smaller than but generally similar to dwarves. Gnomes are known to be expert miners, and it is known that a secret underground mine complex built by this race exists within the Mazes of Menace, filled with both riches and danger.

Humans are by far the most common race of the surface world, and are thus the norm by which other races are often compared. Although they have no special abilities, they can succeed in any role.

Orcs are a cruel and barbaric race that hate every living thing (including other orcs). Above all others, Orcs hate Elves with a passion unequalled, and will go out of their way to kill one at any opportunity. The armor and weapons fashioned by the Orcs are typically of inferior quality.

3 What do all those things on the screen mean?

On the screen is kept a map of where you have been and what you have seen on the current dungeon level; as you explore more of the level, it appears on the screen in front of you.

When *NetHack*'s ancestor *rogue* first appeared, its screen orientation was almost unique among computer fantasy games. Since then, screen orientation has become the norm rather than the exception; *NetHack* continues this fine tradition. Unlike text adventure games that accept commands in pseudo-English sentences and explain the results in words, *NetHack* commands are all one or two keystrokes and the results are displayed graphically on the screen. A minimum screen size of 24 lines by 80 columns is recommended; if the screen is larger, only a 21×80 section will be used for the map.

NetHack can even be played by blind players, with the assistance of Braille readers or speech synthesisers. Instructions for configuring *NetHack* for the blind are included later in this document.

NetHack generates a new dungeon every time you play it; even the authors still find it an entertaining and exciting game despite having won several times.

NetHack offers a variety of display options. The options available to you will vary from port to port, depending on the capabilities of your hardware and software, and whether various compile-time options were enabled when your executable was created. The three possible display options are: a monochrome character interface, a color character interface, and a graphical interface using small pictures called tiles. The two character interfaces allow fonts with other characters to be substituted, but the default assignments use standard ASCII characters to represent everything. There is no difference between the various display options with respect to game play. Because we cannot reproduce the tiles or colors in the Guidebook, and because it is common to all ports, we will use the default ASCII characters from the monochrome character display when referring to things you might see on the screen during your game. In order to understand what is going on in *NetHack*, first you must understand what *NetHack* is doing with the screen. The *NetHack* screen replaces the "You see ..." descriptions of text adventure games. Figure 1 is a sample of what a *NetHack* screen might look like. The way the screen looks for you depends on your platform.

```
The bat bites!

        ------
        |....|    ----------
        |.<..|####...@...$.|
        |....-#   |...B....+
        |....|    |.d......|
        ------    -------|--



Player the Rambler     St:12 Dx:7 Co:18 In:11 Wi:9 Ch:15  Neutral
Dlvl:1  $:0  HP:9(12) Pw:3(3) AC:10 Exp:1/19 T:257 Weak
```

Figure 1

The status lines (bottom)

The bottom two lines of the screen contain several cryptic pieces of information describing your current status. If either status line becomes longer than the width of the screen, you might not see all of it. Here are explanations of what the various status items mean (though your configuration may not have all the status items listed below):

Rank Your character's name and professional ranking (based on the experience level, see below).

Strength A measure of your character's strength; one of your six basic attributes. A human character's attributes can range from 3 to 18 inclusive; non-humans may exceed these limits (occasionally you may get super-strengths of the form 18/xx, and magic can also cause attributes to exceed the normal limits). The higher your strength, the stronger you are. Strength affects how successfully you perform physical tasks, how much damage you do in combat, and how much loot you can carry.

Dexterity Dexterity affects your chances to hit in combat, to avoid traps, and do other tasks requiring agility or manipulation of objects.

Constitution Constitution affects your ability to recover from injuries and other strains on your stamina.

Intelligence Intelligence affects your ability to cast spells and read spellbooks.

Wisdom Wisdom comes from your practical experience (especially when dealing with magic). It affects your magical energy.

Charisma Charisma affects how certain creatures react toward you. In particular, it can affect the prices shopkeepers offer you.

Alignment *Lawful*, *Neutral* or *Chaotic*. Often, Lawful is taken as good and Chaotic is evil, but legal and ethical do not always coincide. Your alignment influences how other monsters react toward you. Monsters of a like alignment are more likely to be non-aggressive, while those of an opposing alignment are more likely to be seriously offended at your presence.

Dungeon Level How deep you are in the dungeon. You start at level one and the number increases as you go deeper into the dungeon. Some levels are special, and are identified by a name and not a number. The Amulet of Yendor is reputed to be somewhere beneath the twentieth level.

Gold The number of gold pieces you are openly carrying. Gold which you have concealed in containers is not counted.

Hit Points Your current and maximum hit points. Hit points indicate how much damage you can take before you die. The more you get hit in a fight, the lower they get. You can regain hit points by resting, or by using certain magical items or spells. The number in parentheses is the maximum number your hit points can reach.

Power Spell points. This tells you how much mystic energy (*mana*) you have available for spell casting. Again, resting will regenerate the amount available.

Armor Class A measure of how effectively your armor stops blows from unfriendly creatures. The lower this number is, the more effective the armor; it is quite possible to have negative armor class.

Experience Your current experience level and experience points. As you adventure, you gain experience points. At certain experience point totals, you gain an experience level. The more experienced you are, the better you fight and withstand magical attacks. Many dungeons show only your experience level here.

Time The number of turns elapsed so far, displayed if you have the *time* option set.

Hunger Status Your current hunger status, ranging from *Satiated* down to *Fainting*. If your hunger status is normal, it is not displayed. Additional status flags may appear after the hunger status: *Conf* when you're confused, *FoodPois* or *Ill* when sick, *Blind* when you can't see, *Stun* when stunned, and *Hallu* when hallucinating.

The message line (top)

The top line of the screen is reserved for messages that describe things that are impossible to represent visually. If you see a "`--More--`" on the top line, this means that *NetHack* has another message to display on the screen, but it wants to make certain that you've read the one that is there first. To read the next message, just press the space bar.

The map (rest of the screen)

The rest of the screen is the map of the level as you have explored it so far. Each symbol on the screen represents something. You can set various graphics options to change some of the symbols the game uses; otherwise, the game will use default symbols. Here is a list of what the default symbols mean:

`-` and `|` — The walls of a room, or an open door. Or a grave (`|`).

`.` — The floor of a room, ice, or a doorless doorway.

`#` — A corridor, or iron bars, or a tree, or possibly a kitchen sink (if your dungeon has sinks), or a drawbridge.

`>` — Stairs down: a way to the next level.

`<` — Stairs up: a way to the previous level.

`+` — A closed door, or a spellbook containing a spell you may be able to learn.

`@` — Your character or a human.

`$` — A pile of gold.

`^` — A trap (once you have detected it).

`)` — A weapon.

`[` — A suit or piece of armor.

`%` — Something edible (not necessarily healthy).

`?` — A scroll.

`/` — A wand.

`=` — A ring.

`!` — A potion.

`(` — A useful item (pick-axe, key, lamp ...).

`"` — An amulet or a spider web.

`*` — A gem or rock (possibly valuable, possibly worthless).

`` ` `` — A boulder or statue.

`0` — An iron ball.

`_` — An altar, or an iron chain.

`{` — A fountain.

`}` — A pool of water or moat or a pool of lava.

`\` — An opulent throne.

`a-zA-Z` & other symbols — Letters and certain other symbols represent the various inhabitants of the Mazes of Menace. Watch out, they can be nasty and vicious. Sometimes, however, they can be helpful.

`I` — This marks the last known location of an invisible or otherwise unseen monster. Note that the monster could have moved. The 'F' and 'm' commands may be useful here.

You need not memorize all these symbols; you can ask the game what any symbol represents with the '`/`' command (see the next section for more info).

4 Commands

Commands are initiated by typing one or two characters. Some commands, like “`search`”, do not require that any more information be collected by *NetHack*. Other commands might require additional information, for example a direction, or an object to be used. For those commands that require additional information, *NetHack* will present you with either a menu of choices, or with a command line prompt requesting information. Which you are presented with will depend chiefly on how you have set the ‘*menustyle*’ option.

For example, a common question in the form “`What do you want to use?  [a-zA-Z ?*]`”, asks you to choose an object you are carrying. Here, “`a-zA-Z`” are the inventory letters of your possible choices. Typing ‘`?`’ gives you an inventory list of these items, so you can see what each letter refers to. In this example, there is also a ‘`*`’ indicating that you may choose an object not on the list, if you wanted to use something unexpected. Typing a ‘`*`’ lists your entire inventory, so you can see the inventory letters of every object you’re carrying. Finally, if you change your mind and decide you don’t want to do this command after all, you can press the ‘ESC’ key to abort the command.

You can put a number before some commands to repeat them that many times; for example, “`10s`” will search ten times. If you have the *number_pad* option set, you must type ‘`n`’ to prefix a count, so the example above would be typed “`n10s`” instead. Commands for which counts make no sense ignore them. In addition, movement commands can be prefixed for greater control (see below). To cancel a count or a prefix, press the ‘ESC’ key.

The list of commands is rather long, but it can be read at any time during the game through the ‘`?`’ command, which accesses a menu of helpful texts. Here are the commands for your reference:

`?` — Help menu: display one of several help texts available.

`/` — Tell what a symbol represents. You may choose to specify a location or type a symbol (or even a whole word) to explain. Specifying a location is done by moving the cursor to a particular spot on the map and then pressing one of ‘`.`’, ‘`,`’, ‘`;`’, or ‘`:`’. ‘`.`’ will explain the symbol at the chosen location, conditionally check for “`More info?`” depending upon whether the *help* option is on, and then you will be asked to pick another location; ‘`,`’ will explain the symbol but skip any additional information; ‘`;`’ will skip additional info and also not bother asking you to choose another location to examine; ‘`:`’ will show additional info, if any, without asking for confirmation. When picking a location, pressing the `ESC` key will terminate this command, or pressing ‘`?`’ will give a brief reminder about how it works.
Specifying a name rather than a location always gives any additional information available about that name.

`&` — Tell what a command does.

`<` — Go up to the previous level (if you are on a staircase or ladder).

`>` — Go down to the next level (if you are on a staircase or ladder).

`[yuhjklbn]` — Go one step in the direction indicated (see Figure 2). If you sense or remember a monster there, you will fight the monster instead. Only these one-step movement commands cause you to fight monsters; the others (below) are “safe.”

```
y  k  u           7  8  9
 \ | /             \ | /
h- . -l           4- . -6
 / | \             / | \
b  j  n           1  2  3
```

(if *number_pad* set)

Figure 2

`[YUHJKLBN]` — Go in that direction until you hit a wall or run into something.

`m[yuhjklbn]` Prefix: move without picking up objects or fighting (even if you remember a monster there)

`F[yuhjklbn]` Prefix: fight a monster (even if you only guess one is there)

`M[yuhjklbn]` Prefix: Move far, no pickup.

`g[yuhjklbn]` Prefix: Move until something interesting is found.

`G[yuhjklbn]` or `<CONTROL->[yuhjklbn]` Prefix: Same as '`g`', but forking of corridors is not considered interesting.

`_` Travel to a map location via a shortest-path algorithm. The shortest path is computed over map locations the hero knows about (e.g. seen or previously traversed). If there is no known path, a guess is made instead. Stops on most of the same conditions as the 'G' command, but without picking up objects, similar to the 'M' command. For ports with mouse support, the command is also invoked when a mouse-click takes place on a location other than the current position.

`.` Rest, do nothing for one turn.

`a` Apply (use) a tool (pick-axe, key, lamp . . .).

`A` Remove one or more worn items, such as armor. Use '`T`' (take off) to take off only one piece of armor or '`R`' (remove) to take off only one accessory.

`^A` Redo the previous command.

`c` Close a door.

`C` Call (name) an individual monster.

`^C` Panic button. Quit the game.

`d` Drop something.
`d7a` — drop seven items of object *a*.

`D` Drop several things. In answer to the question "`What kinds of things do you want to drop? [!%= BUCXaium]`" you should type zero or more object symbols possibly followed by '`a`' and/or '`i`' and/or '`u`' and/or '`m`'. In addition, one or more of the blessed/uncursed/cursed groups may be typed.
`DB` — drop all objects known to be blessed.
`DU` — drop all objects known to be uncursed.
`DC` — drop all objects known to be cursed.
`DX` — drop all objects of unknown B/U/C status.
`Da` — drop all objects, without asking for confirmation.
`Di` — examine your inventory before dropping anything.
`Du` — drop only unpaid objects (when in a shop).
`Dm` — use a menu to pick which object(s) to drop.
`D%u` — drop only unpaid food.

`^D` Kick something (usually a door).

`e` Eat food.

`E` Engrave a message on the floor. Engraving the word "`Elbereth`" will cause most monsters to not attack you hand-to-hand (but if you attack, you will rub it out); this is often useful to give yourself a breather. (This feature may be compiled out of the game, so your version might not have it.)
`E-` — write in the dust with your fingers.

`f` Fire one of the objects placed in your quiver. You may select ammunition with a previous '`Q`' command, or let the computer pick something appropriate if *autoquiver* is true.

`i` List your inventory (everything you're carrying).

`I`	List selected parts of your inventory. `I*` — list all gems in inventory; `Iu` — list all unpaid items; `Ix` — list all used up items that are on your shopping bill; `I$` — count your money.
`o`	Open a door.
`O`	Set options. A menu showing the current option values will be displayed. You can change most values simply by selecting the menu entry for the given option (ie, by typing its letter or clicking upon it, depending on your user interface). For the non-boolean choices, a further menu or prompt will appear once you've closed this menu. The available options are listed later in this Guidebook. Options are usually set before the game rather than with the '`O`' command; see the section on options below.
`p`	Pay your shopping bill.
`P`	Put on a ring or other accessory (amulet, blindfold).
`^P`	Repeat previous message. Subsequent `^P`'s repeat earlier messages. The behavior can be varied via the msg˙window option.
`q`	Quaff (drink) something (potion, water, etc).
`Q`	Select an object for your quiver. You can then throw this using the 'f' command. (In versions prior to 3.3 this was the command to quit the game, which has now been moved to '`#quit`'.)
`r`	Read a scroll or spellbook.
`R`	Remove an accessory (ring, amulet, etc).
`^R`	Redraw the screen.
`s`	Search for secret doors and traps around you. It usually takes several tries to find something.
`S`	Save (and suspend) the game. The game will be restored automatically the next time you play.
`t`	Throw an object or shoot a projectile.
`T`	Take off armor.
`^T`	Teleport, if you have the ability.
`v`	Display version number.
`V`	Display the game history.
`w`	Wield weapon. `w-` — wield nothing, use your bare hands.
`W`	Wear armor.
`x`	Exchange your wielded weapon with the item in your alternate weapon slot. The latter is used as your secondary weapon when engaging in two-weapon combat. Note that if one of these slots is empty, the exchange still takes place.
`X`	Enter explore (discovery) mode, explained in its own section later.
`^X`	Display your name, role, race, gender, and alignment as well as the various deities in your game.
`z`	Zap a wand. To aim at yourself, use '`.`' for the direction.
`Z`	Zap (cast) a spell. To cast at yourself, use '`.`' for the direction.
`^Z`	Suspend the game (UNIX versions with job control only).
`:`	Look at what is here.

`;` Show what type of thing a visible symbol corresponds to.

`,` Pick up some things. May be preceded by ‘`m`’ to force a selection menu.

`@` Toggle the *autopickup* option on and off.

`^` Ask for the type of a trap you found earlier.

`)` Tell what weapon you are wielding.

`[` Tell what armor you are wearing.

`=` Tell what rings you are wearing.

`"` Tell what amulet you are wearing.

`(` Tell what tools you are using.

`*` Tell what equipment you are using; combines the preceding five type-specific commands into one.

`$` Count your gold pieces.

`+` List the spells you know. Using this command, you can also rearrange the order in which your spells are listed. They are shown via a menu, and if you select a spell in that menu, you'll be re-prompted for another spell to swap places with it, and then have opportunity to make further exchanges.

\ Show what types of objects have been discovered.

`!` Escape to a shell.

`#` Perform an extended command. As you can see, the authors of *NetHack* used up all the letters, so this is a way to introduce the less frequently used commands. What extended commands are available depends on what features the game was compiled with.

`#adjust` Adjust inventory letters (most useful when the *fixinv* option is “on”).

`#chat` Talk to someone.

`#conduct` List which challenges you have adhered to. See the section below entitled “Conduct” for details.

`#dip` Dip an object into something.

`#enhance` Advance or check weapons and spell skills.

`#force` Force a lock.

`#invoke` Invoke an object's special powers.

`#jump` Jump to another location.

`#loot` Loot a box or bag on the floor beneath you, or the saddle from a horse standing next to you.

`#monster` Use a monster's special ability (when polymorphed into monster form).

`#name` Name an item or type of object.

`#offer` Offer a sacrifice to the gods.

`#pray` Pray to the gods for help.

`#quit` Quit the program without saving your game.

`#ride` Ride (or stop riding) a monster.

`#rub` Rub a lamp or a stone.

`#sit` Sit down.

`#turn` Turn undead.

`#twoweapon` Toggle two-weapon combat on or off. Note that you must use suitable weapons for this type of combat, or it will be automatically turned off.

#untrap	Untrap something (trap, door, or chest).
#version	Print compile time options for this version of *NetHack*.
#wipe	Wipe off your face.
#?	Help menu: get the list of available extended commands.

If your keyboard has a meta key (which, when pressed in combination with another key, modifies it by setting the 'meta' [8th, or 'high'] bit), you can invoke many extended commands by meta-ing the first letter of the command. In *NT, OS/2,* and *PC NetHack*, the 'Alt' key can be used in this fashion.

M-?	#? (not supported by all platforms)
M-2	#twoweapon (unless the *number_pad* option is enabled)
M-a	#adjust
M-c	#chat
M-d	#dip
M-e	#enhance
M-f	#force
M-i	#invoke
M-j	#jump
M-l	#loot
M-m	#monster
M-n	#name
M-o	#offer
M-p	#pray
M-q	#quit
M-r	#rub
M-s	#sit
M-t	#turn
M-u	#untrap
M-v	#version
M-w	#wipe

If the *number_pad* option is on, some additional letter commands are available:

h	Help menu: display one of several help texts available, like "?".
j	Jump to another location. Same as "#jump" or "M-j".
k	Kick something (usually a door). Same as '^D'.
l	Loot a box or bag on the floor beneath you, or the saddle from a horse standing next to you. Same as "#loot" or "M-l".
N	Name an object or type of object. Same as "#name" or "M-n".
u	Untrap a trap, door, or chest. Same as "#untrap" or "M-u".

5 Rooms and corridors

Rooms and corridors in the dungeon are either lit or dark. Any lit areas within your line of sight will be displayed; dark areas are only displayed if they are within one space of you. Walls and corridors remain on the map as you explore them.

Secret corridors are hidden. You can find them with the 's' (search) command.

Doorways

Doorways connect rooms and corridors. Some doorways have no doors; you can walk right through. Others have doors in them, which may be open, closed, or locked. To open a closed door, use the ‘`o`’ (open) command; to close it again, use the ‘`c`’ (close) command.

You can get through a locked door by using a tool to pick the lock with the ‘`a`’ (apply) command, or by kicking it open with the ‘`^D`’ (kick) command.

Open doors cannot be entered diagonally; you must approach them straight on, horizontally or vertically. Doorways without doors are not restricted in this fashion.

Doors can be useful for shutting out monsters. Most monsters cannot open doors, although a few don’t need to (ex. ghosts can walk through doors).

Secret doors are hidden. You can find them with the ‘`s`’ (search) command. Once found they are in all ways equivalent to normal doors.

Traps (‘`^`’)

There are traps throughout the dungeon to snare the unwary delver. For example, you may suddenly fall into a pit and be stuck for a few turns trying to climb out. Traps don’t appear on your map until you see one triggered by moving onto it, see something fall into it, or you discover it with the ‘`s`’ (search) command. Monsters can fall prey to traps, too, which can be a very useful defensive strategy.

There is a special pre-mapped branch of the dungeon based on the classic computer game “`Sokoban`.” The goal is to push the boulders into the pits or holes. With careful foresight, it is possible to complete all of the levels according to the traditional rules of Sokoban. Some allowances are permitted in case the player gets stuck; however, they will lower your luck.

Stairs (‘<’, ‘>’)

In general, each level in the dungeon will have a staircase going up (‘`<`’) to the previous level and another going down (‘`>`’) to the next level. There are some exceptions though. For instance, fairly early in the dungeon you will find a level with two down staircases, one continuing into the dungeon and the other branching into an area known as the Gnomish Mines. Those mines eventually hit a dead end, so after exploring them (if you choose to do so), you’ll need to climb back up to the main dungeon.

When you traverse a set of stairs, or trigger a trap which sends you to another level, the level you’re leaving will be deactivated and stored in a file on disk. If you’re moving to a previously visited level, it will be loaded from its file on disk and reactivated. If you’re moving to a level which has not yet been visited, it will be created (from scratch for most random levels, from a template for some “special” levels, or loaded from the remains of an earlier game for a “bones” level as briefly described below). Monsters are only active on the current level; those on other levels are essentially placed into stasis.

Ordinarily when you climb a set of stairs, you will arrive on the corresponding staircase at your destination. However, pets (see below) and some other monsters will follow along if they’re close enough when you travel up or down stairs, and occasionally one of these creatures will displace you during the climb. When that occurs, the pet or other monster will arrive on the staircase and you will end up nearby.

Ladders (‘<’, ‘>’)

Ladders serve the same purpose as staircases, and the two types of inter-level connections are nearly indistinguishable during game play.

6 Monsters

Monsters you cannot see are not displayed on the screen. Beware! You may suddenly come upon one in a dark place. Some magic items can help you locate them before they locate you (which some monsters can do very well).

The commands '/' and ';' may be used to obtain information about those monsters who are displayed on the screen. The command 'C' allows you to assign a name to a monster, which may be useful to help distinguish one from another when multiple monsters are present. Assigning a name which is just a space will remove any prior name.

The extended command "#chat" can be used to interact with an adjacent monster. There is no actual dialog (in other words, you don't get to choose what you'll say), but chatting with some monsters such as a shopkeeper or the Oracle of Delphi can produce useful results.

Fighting

If you see a monster and you wish to fight it, just attempt to walk into it. Many monsters you find will mind their own business unless you attack them. Some of them are very dangerous when angered. Remember: discretion is the better part of valor.

If you can't see a monster (if it is invisible, or if you are blinded), the symbol 'I' will be shown when you learn of its presence. If you attempt to walk into it, you will try to fight it just like a monster that you can see; of course, if the monster has moved, you will attack empty air. If you guess that the monster has moved and you don't wish to fight, you can use the 'm' command to move without fighting; likewise, if you don't remember a monster but want to try fighting anyway, you can use the 'F' command.

Your pet

You start the game with a little dog ('d'), cat ('f'), or pony ('u'), which follows you about the dungeon and fights monsters with you. Like you, your pet needs food to survive. It usually feeds itself on fresh carrion and other meats. If you're worried about it or want to train it, you can feed it, too, by throwing it food. A properly trained pet can be very useful under certain circumstances.

Your pet also gains experience from killing monsters, and can grow over time, gaining hit points and doing more damage. Initially, your pet may even be better at killing things than you, which makes pets useful for low-level characters.

Your pet will follow you up and down staircases if it is next to you when you move. Otherwise your pet will be stranded and may become wild. Similarly, when you trigger certain types of traps which alter your location (for instance, a trap door which drops you to a lower dungeon level), any adjacent pet will accompany you and any non-adjacent pet will be left behind. Your pet may trigger such traps itself; you will not be carried along with it even if adjacent at the time.

Steeds

Some types of creatures in the dungeon can actually be ridden if you have the right equipment and skill. Convincing a wild beast to let you saddle it up is difficult to say the least. Many a dungeoneer has had to resort to magic and wizardry in order to forge the alliance. Once you do have the beast under your control however, you can easily climb in and out of the saddle with the '#ride' command. Lead the beast around the dungeon when riding, in the same manner as you would move yourself. It is the beast that you will see displayed on the map.

Riding skill is managed by the '#enhance' command. See the section on Weapon proficiency for more information about that.

Bones levels

You may encounter the shades and corpses of other adventurers (or even former incarnations of yourself!) and their personal effects. Ghosts are hard to kill, but easy to avoid, since they're slow and do little damage. You can plunder the deceased adventurer's possessions; however, they are likely to be cursed. Beware of whatever killed the former player; it is probably still lurking around, gloating over its last victory.

7 Objects

When you find something in the dungeon, it is common to want to pick it up. In *NetHack*, this is accomplished automatically by walking over the object (unless you turn off the *autopickup* option (see below), or move with the '`m`' prefix (see above)), or manually by using the '`,`' command. If you're carrying too many items, *NetHack* will tell you so and you won't be able to pick up anything more. Otherwise, it will add the object(s) to your pack and tell you what you just picked up. As you add items to your inventory, you also add the weight of that object to your load. The amount that you can carry depends on your strength and your constitution. The stronger you are, the less the additional load will affect you. There comes a point, though, when the weight of all of that stuff you are carrying around with you through the dungeon will encumber you. Your reactions will get slower and you'll burn calories faster, requiring food more frequently to cope with it. Eventually, you'll be so overloaded that you'll either have to discard some of what you're carrying or collapse under its weight. NetHack will tell you how badly you have loaded yourself. The symbols 'Burdened', 'Stressed', 'Strained', 'Overtaxed' and 'Overloaded' are displayed on the bottom line display to indicate your condition.

When you pick up an object, it is assigned an inventory letter. Many commands that operate on objects must ask you to find out which object you want to use. When *NetHack* asks you to choose a particular object you are carrying, you are usually presented with a list of inventory letters to choose from (see Commands, above).

Some objects, such as weapons, are easily differentiated. Others, like scrolls and potions, are given descriptions which vary according to type. During a game, any two objects with the same description are the same type. However, the descriptions will vary from game to game.

When you use one of these objects, if its effect is obvious, *NetHack* will remember what it is for you. If its effect isn't extremely obvious, you will be asked what you want to call this type of object so you will recognize it later. You can also use the "`#name`" command for the same purpose at any time, to name all objects of a particular type or just an individual object. When you use "`#name`" on an object which has already been named, specifying a space as the value will remove the prior name instead of assigning a new one.

Curses and Blessings

Any object that you find may be cursed, even if the object is otherwise helpful. The most common effect of a curse is being stuck with (and to) the item. Cursed weapons weld themselves to your hand when wielded, so you cannot unwield them. Any cursed item you wear is not removable by ordinary means. In addition, cursed arms and armor usually, but not always, bear negative enchantments that make them less effective in combat. Other cursed objects may act poorly or detrimentally in other ways.

Objects can also be blessed. Blessed items usually work better or more beneficially than normal uncursed items. For example, a blessed weapon will do more damage against demons.

There are magical means of bestowing or removing curses upon objects, so even if you are stuck with one, you can still have the curse lifted and the item removed. Priests and Priestesses have an innate sensitivity to this property in any object, so they can more easily avoid cursed objects than other character roles.

An item with unknown status will be reported in your inventory with no prefix. An item which you know the state of will be distinguished in your inventory by the presence of the word "cursed", "uncursed" or "blessed" in the description of the item.

Weapons ('`)`')

Given a chance, most monsters in the Mazes of Menace will gratuitously try to kill you. You need weapons for self-defense (killing them first). Without a weapon, you do only 1–2 hit points of damage (plus bonuses, if any). Monk characters are an exception; they normally do much more damage with bare hands than they do with weapons.

There are wielded weapons, like maces and swords, and thrown weapons, like arrows and spears. To hit monsters with a weapon, you must wield it and attack them, or throw it at them. You can simply elect to throw a spear. To shoot an arrow, you should first wield a bow, then throw the arrow. Crossbows shoot crossbow bolts. Slings hurl rocks and (other) stones (like gems).

Enchanted weapons have a "plus" (or "to hit enhancement" which can be either positive or negative) that adds to your chance to hit and the damage you do to a monster. The only way to determine a weapon's enchantment is to have it magically identified somehow. Most weapons are subject to some type of damage like rust. Such "erosion" damage can be repaired.

The chance that an attack will successfully hit a monster, and the amount of damage such a hit will do, depends upon many factors. Among them are: type of weapon, quality of weapon (enchantment and/or erosion), experience level, strength, dexterity, encumbrance, and proficiency (see below). The monster's armor class—a general defense rating, not necessarily due to wearing of armor—is a factor too; also, some monsters are particularly vulnerable to certain types of weapons.

Many weapons can be wielded in one hand; some require both hands. When wielding a two-handed weapon, you can not wear a shield, and vice versa. When wielding a one-handed weapon, you can have another weapon ready to use by setting things up with the '`x`' command, which exchanges your primary (the one being wielded) and alternate weapons. And if you have proficiency in the "two weapon combat" skill, you may wield both weapons simultaneously as primary and secondary; use the '`#twoweapon`' extended command to engage or disengage that. Only some types of characters (barbarians, for instance) have the necessary skill available. Even with that skill, using two weapons at once incurs a penalty in the chance to hit your target compared to using just one weapon at a time.

There might be times when you'd rather not wield any weapon at all. To accomplish that, wield '`-`', or else use the '`A`' command which allows you to unwield the current weapon in addition to taking off other worn items.

Those of you in the audience who are AD&D players, be aware that each weapon which existed in AD&D does roughly the same damage to monsters in *NetHack*. Some of the more obscure weapons (such as the *aklys*, *lucern hammer*, and *bec-de-corbin*) are defined in an appendix to *Unearthed Arcana*, an AD&D supplement.

The commands to use weapons are '`w`' (wield), '`t`' (throw), '`f`' (fire, an alternative way of throwing), '`Q`' (quiver), '`x`' (exchange), '`#twoweapon`', and '`#enhance`' (see below).

Throwing and shooting

You can throw just about anything via the '`t`' command. It will prompt for the item to throw; picking '`?`' will list things in your inventory which are considered likely to be thrown, or picking '`*`' will list your entire inventory. After you've chosen what to throw, you will be prompted for a direction rather than for a specific target. The distance something can be thrown depends mainly on the type of object and your strength. Arrows can be thrown by hand, but can be thrown much farther and will be more likely to hit when thrown while you are wielding a bow.

You can simplify the throwing operation by using the '`Q`' command to select your preferred "missile", then using the '`f`' command to throw it. You'll be prompted for a direction as above, but you don't have to specify which item to throw each time you use '`f`'. There is also an option, *autoquiver*, which has *NetHack* choose another item to automatically fill your quiver when the inventory slot used for '`Q`' runs out.

Some characters have the ability to fire a volley of multiple items in a single turn. Knowing how to load several rounds of ammunition at once—or hold several missiles in your hand—and still hit a target is not an easy task. Rangers are among those who are adept at this task, as are those with a high level of proficiency in the relevant weapon skill (in bow skill if you're wielding one to shoot arrows, in crossbow skill if you're wielding one to shoot bolts, or in sling skill if you're wielding one to shoot stones). The number of items that the character has a chance to fire varies from turn to turn. You can explicitly limit the number of shots by using a numeric prefix before the '`t`' or '`f`' command. For example, "`2f`" (or "`n2f`" if using *number_pad* mode) would ensure that at most 2 arrows are shot even if you could have fired 3. If you specify a larger number than would have been shot ("4f" in this example), you'll just end up shooting the same number (3, here) as if no limit had been specified. Once the volley is in motion, all of the items will travel in the same direction; if the first ones kill a monster, the others can still continue beyond that spot.

Weapon proficiency

You will have varying degrees of skill in the weapons available. Weapon proficiency, or weapon skills, affect how well you can use particular types of weapons, and you'll be able to improve your skills as you progress through a game, depending on your role, your experience level, and use of the weapons.

For the purposes of proficiency, weapons have been divided up into various groups such as daggers, broadswords, and polearms. Each role has a limit on what level of proficiency a character can achieve for each group. For instance, wizards can become highly skilled in daggers or staves but not in swords or bows.

The '#enhance' extended command is used to review current weapons proficiency (also spell proficiency) and to choose which skill(s) to improve when you've used one or more skills enough to become eligible to do so. The skill rankings are "none" (sometimes also referred to as "restricted", because you won't be able to advance), "unskilled", "basic", "skilled", and "expert". Restricted skills simply will not appear in the list shown by '#enhance'. (Divine intervention might unrestrict a particular skill, in which case it will start at unskilled and be limited to basic.) Some characters can enhance their barehanded combat or martial arts skill beyond expert to "master" or "grand master".

Use of a weapon in which you're restricted or unskilled will incur a modest penalty in the chance to hit a monster and also in the amount of damage done when you do hit; at basic level, there is no penalty or bonus; at skilled level, you receive a modest bonus in the chance to hit and amount of damage done; at expert level, the bonus is higher. A successful hit has a chance to boost your training towards the next skill level (unless you've already reached the limit for this skill). Once such training reaches the threshold for that next level, you'll be told that you feel more confident in your skills. At that point you can use '#enhance' to increase one or more skills. Such skills are not increased automatically because there is a limit to your total overall skills, so you need to actively choose which skills to enhance and which to ignore.

Armor ('[')

Lots of unfriendly things lurk about; you need armor to protect yourself from their blows. Some types of armor offer better protection than others. Your armor class is a measure of this protection. Armor class (AC) is measured as in AD&D, with 10 being the equivalent of no armor, and lower numbers meaning better armor. Each suit of armor which exists in AD&D gives the same protection in *NetHack*. Here is an (incomplete) list of the armor classes provided by various suits of armor:

dragon scale mail	1	plate mail	3
crystal plate mail	3	bronze plate mail	4
splint mail	4	banded mail	4
dwarvish mithril-coat	4	elven mithril-coat	5
chain mail	5	orcish chain mail	6
scale mail	6	studded leather armor	7
ring mail	7	orcish ring mail	8
leather armor	8	leather jacket	9
no armor	10		

You can also wear other pieces of armor (ex. helmets, boots, shields, cloaks) to lower your armor class even further, but you can only wear one item of each category (one suit of armor, one cloak, one helmet, one shield, and so on) at a time.

If a piece of armor is enchanted, its armor protection will be better (or worse) than normal, and its "plus" (or minus) will subtract from your armor class. For example, a +1 chain mail would give you better protection than normal chain mail, lowering your armor class one unit further to 4. When you put on a piece of armor, you immediately find out the armor class and any "plusses" it provides. Cursed pieces of armor usually have negative enchantments (minuses) in addition to being unremovable.

Many types of armor are subject to some kind of damage like rust. Such damage can be repaired. Some types of armor may inhibit spell casting.

The commands to use armor are '`W`' (wear) and '`T`' (take off). The '`A`' command can also be used to take off armor as well as other worn items.

Food ('%')

Food is necessary to survive. If you go too long without eating you will faint, and eventually die of starvation. Some types of food will spoil, and become unhealthy to eat, if not protected. Food stored in ice boxes or tins ("cans") will usually stay fresh, but ice boxes are heavy, and tins take a while to open.

When you kill monsters, they usually leave corpses which are also "food." Many, but not all, of these are edible; some also give you special powers when you eat them. A good rule of thumb is "you are what you eat."

Some character roles and some monsters are vegetarian. Vegetarian monsters will typically never eat animal corpses, while vegetarian players can, but with some rather unpleasant side-effects.

You can name one food item after something you like to eat with the *fruit* option.

The command to eat food is '`e`'.

Scrolls ('?')

Scrolls are labeled with various titles, probably chosen by ancient wizards for their amusement value (ex. "READ ME," or "THANX MAUD" backwards). Scrolls disappear after you read them (except for blank ones, without magic spells on them).

One of the most useful of these is the *scroll of identify*, which can be used to determine what another object is, whether it is cursed or blessed, and how many uses it has left. Some objects of subtle enchantment are difficult to identify without these.

A mail daemon may run up and deliver mail to you as a *scroll of mail* (on versions compiled with this feature). To use this feature on versions where *NetHack* mail delivery is triggered by electronic mail appearing in your system mailbox, you must let *NetHack* know where to look for new mail by setting the "MAIL" environment variable to the file name of your mailbox. You may also want to set the "MAILREADER" environment variable to the file name of your favorite reader, so *NetHack* can shell to it when you read the scroll. On versions of *NetHack* where mail is randomly generated internal to the game, these environment variables are ignored. You can disable the mail daemon by turning off the *mail* option.

The command to read a scroll is '`r`'.

Potions ('!')

Potions are distinguished by the color of the liquid inside the flask. They disappear after you quaff them.

Clear potions are potions of water. Sometimes these are blessed or cursed, resulting in holy or unholy water. Holy water is the bane of the undead, so potions of holy water are good things to throw ('`t`') at them. It is also sometimes very useful to dip ("`#dip`") an object into a potion.

The command to drink a potion is '`q`' (quaff).

Wands ('/')

Magic wands usually have multiple magical charges. Some wands are directional—you must give a direction in which to zap them. You can also zap them at yourself (just give a '`.`' or '`s`' for the direction).

Be warned, however, for this is often unwise. Other wands are nondirectional—they don't require a direction. The number of charges in a wand is random and decreases by one whenever you use it.

When the number of charges left in a wand becomes zero, attempts to use the wand will usually result in nothing happening. Occasionally, however, it may be possible to squeeze the last few mana points from an otherwise spent wand, destroying it in the process. A wand may be recharged by using suitable magic, but doing so runs the risk of causing it to explode. The chance for such an explosion starts out very small and increases each time the wand is recharged.

In a truly desperate situation, when your back is up against the wall, you might decide to go for broke and break your wand. This is not for the faint of heart. Doing so will almost certainly cause a catastrophic release of magical energies.

When you have fully identified a particular wand, inventory display will include additional information in parentheses: the number of times it has been recharged followed by a colon and then by its current number of charges. A current charge count of `-1` is a special case indicating that the wand has been cancelled.

The command to use a wand is '`z`' (zap). To break one, use the '`a`' (apply) command.

Rings ('=')

Rings are very useful items, since they are relatively permanent magic, unlike the usually fleeting effects of potions, scrolls, and wands.

Putting on a ring activates its magic. You can wear only two rings, one on each ring finger.

Most rings also cause you to grow hungry more rapidly, the rate varying with the type of ring.

The commands to use rings are '`P`' (put on) and '`R`' (remove).

Spellbooks ('+')

Spellbooks are tomes of mighty magic. When studied with the '`r`' (read) command, they transfer to the reader the knowledge of a spell (and therefore eventually become unreadable) — unless the attempt backfires. Reading a cursed spellbook or one with mystic runes beyond your ken can be harmful to your health!

A spell (even when learned) can also backfire when you cast it. If you attempt to cast a spell well above your experience level, or if you have little skill with the appropriate spell type, or cast it at a time when your luck is particularly bad, you can end up wasting both the energy and the time required in casting.

Casting a spell calls forth magical energies and focuses them with your naked mind. Some of the magical energy released comes from within you, and casting several spells in a row may tire you. Casting of spells also requires practice. With practice, your skill in each category of spell casting will improve. Over time, however, your memory of each spell will dim, and you will need to relearn it.

Some spells are directional—you must give a direction in which to cast them. You can also cast them at yourself (just give a '`.`' or '`s`' for the direction). Be warned, however, for this is often unwise. Other spells are nondirectional—they don't require a direction.

Just as weapons are divided into groups in which a character can become proficient (to varying degrees), spells are similarly grouped. Successfully casting a spell exercises the skill group; sufficient skill may increase the potency of the spell and reduce the risk of spell failure. Skill slots are shared with weapons skills. (See also the section on "Weapon proficiency".)

Casting a spell also requires flexible movement, and wearing various types of armor may interfere with that.

The command to read a spellbook is the same as for scrolls, '`r`' (read). The '`+`' command lists your current spells, their levels, categories, and chances for failure. The '`Z`' (cast) command casts a spell.

Tools (‘(’)

Tools are miscellaneous objects with various purposes. Some tools have a limited number of uses, akin to wand charges. For example, lamps burn out after a while. Other tools are containers, which objects can be placed into or taken out of.

The command to use tools is ‘`a`’ (apply).

Containers

You may encounter bags, boxes, and chests in your travels. A tool of this sort can be opened with the “`#loot`” extended command when you are standing on top of it (that is, on the same floor spot), or with the ‘`a`’ (apply) command when you are carrying it. However, chests are often locked, and are in any case unwieldy objects. You must set one down before unlocking it by using a key or lock-picking tool with the ‘`a`’ (apply) command, by kicking it with the ‘`^D`’ command, or by using a weapon to force the lock with the “`#force`” extended command.

Some chests are trapped, causing nasty things to happen when you unlock or open them. You can check for and try to deactivate traps with the “`#untrap`” extended command.

Amulets (‘"’)

Amulets are very similar to rings, and often more powerful. Like rings, amulets have various magical properties, some beneficial, some harmful, which are activated by putting them on.

Only one amulet may be worn at a time, around your neck.

The commands to use amulets are the same as for rings, ‘`P`’ (put on) and ‘`R`’ (remove).

Gems (‘*’)

Some gems are valuable, and can be sold for a lot of gold. They are also a far more efficient way of carrying your riches. Valuable gems increase your score if you bring them with you when you exit.

Other small rocks are also categorized as gems, but they are much less valuable. All rocks, however, can be used as projectile weapons (if you have a sling). In the most desperate of cases, you can still throw them by hand.

Large rocks (‘`’)

Statues and boulders are not particularly useful, and are generally heavy. It is rumored that some statues are not what they seem.

Very large humanoids (giants and their ilk) have been known to use boulders as weapons.

Gold (‘$’)

Gold adds to your score, and you can buy things in shops with it. There are a number of monsters in the dungeon that may be influenced by the amount of gold you are carrying (shopkeepers aside).

8 Conduct

As if winning *NetHack* were not difficult enough, certain players seek to challenge themselves by imposing restrictions on the way they play the game. The game automatically tracks some of these challenges, which can be checked at any time with the `#conduct` command or at the end of the game. When you perform an action which breaks a challenge, it will no longer be listed. This gives players extra “bragging rights” for winning the game with these challenges. Note that it is perfectly acceptable to win the game without resorting to these restrictions and that it is unusual for players to adhere to challenges the first time they win the game.

Several of the challenges are related to eating behavior. The most difficult of these is the foodless challenge. Although creatures can survive long periods of time without food, there is a physiological need for water; thus there is no restriction on drinking beverages, even if they provide some minor food benefits. Calling upon your god for help with starvation does not violate any food challenges either.

A strict vegan diet is one which avoids any food derived from animals. The primary source of nutrition is fruits and vegetables. The corpses and tins of blobs ('b'), jellies ('j'), and fungi ('F') are also considered to be vegetable matter. Certain human food is prepared without animal products; namely, lembas wafers, cram rations, food rations (gunyoki), K-rations, and C-rations. Metal or another normally indigestible material eaten while polymorphed into a creature that can digest it is also considered vegan food. Note however that eating such items still counts against foodless conduct.

Vegetarians do not eat animals; however, they are less selective about eating animal byproducts than vegans. In addition to the vegan items listed above, they may eat any kind of pudding ('P') other than the black puddings, eggs and food made from eggs (fortune cookies and pancakes), food made with milk (cream pies and candy bars), and lumps of royal jelly. Monks are expected to observe a vegetarian diet.

Eating any kind of meat violates the vegetarian, vegan, and foodless conducts. This includes tripe rations, the corpses or tins of any monsters not mentioned above, and the various other chunks of meat found in the dungeon. Swallowing and digesting a monster while polymorphed is treated as if you ate the creature's corpse. Eating leather, dragon hide, or bone items while polymorphed into a creature that can digest it, or eating monster brains while polymorphed into a mind flayer, is considered eating an animal, although wax is only an animal byproduct.

Regardless of conduct, there will be some items which are indigestible, and others which are hazardous to eat. Using a swallow-and-digest attack against a monster is equivalent to eating the monster's corpse. Please note that the term "vegan" is used here only in the context of diet. You are still free to choose not to use or wear items derived from animals (e.g. leather, dragon hide, bone, horns, coral), but the game will not keep track of this for you. Also note that "milky" potions may be a translucent white, but they do not contain milk, so they are compatible with a vegan diet. Slime molds or player-defined "fruits", although they could be anything from "cherries" to "pork chops", are also assumed to be vegan.

An atheist is one who rejects religion. This means that you cannot `#pray`, `#offer` sacrifices to any god, `#turn` undead, or `#chat` with a priest. Particularly selective readers may argue that playing Monk or Priest characters should violate this conduct; that is a choice left to the player. Offering the Amulet of Yendor to your god is necessary to win the game and is not counted against this conduct. You are also not penalized for being spoken to by an angry god, priest(ess), or other religious figure; a true atheist would hear the words but attach no special meaning to them.

Most players fight with a wielded weapon (or tool intended to be wielded as a weapon). Another challenge is to win the game without using such a wielded weapon. You are still permitted to throw, fire, and kick weapons; use a wand, spell, or other type of item; or fight with your hands and feet.

In *NetHack*, a pacifist refuses to cause the death of any other monster (i.e. if you would get experience for the death). This is a particularly difficult challenge, although it is still possible to gain experience by other means.

An illiterate character cannot read or write. This includes reading a scroll, spellbook, fortune cookie message, or t-shirt; writing a scroll; or making an engraving of anything other than a single "x" (the traditional signature of an illiterate person). Reading an engraving, or any item that is absolutely necessary to win the game, is not counted against this conduct. The identity of scrolls and spellbooks (and knowledge of spells) in your starting inventory is assumed to be learned from your teachers prior to the start of the game and isn't counted.

There are several other challenges tracked by the game. It is possible to eliminate one or more species of monsters by genocide; playing without this feature is considered a challenge. When you game offers you an opportunity to genocide monsters, you may respond with the monster type "none" if you want to decline. You can change the form of an item into another item of the same type ("polypiling") or the form of your own body into another creature ("polyself") by wand, spell, or potion of polymorph; avoiding these effects are each considered challenges. Polymorphing monsters, including pets, does not break either of these challenges. Finally, you may sometimes receive wishes; a game without an attempt to wish for any items is a challenge, as is a game without wishing for an artifact (even if the artifact immediately disappears). When the game offers you an opportunity to make a wish for an item, you may choose "nothing" if you want to decline.

9 Options

Due to variations in personal tastes and conceptions of how *NetHack* should do things, there are options you can set to change how *NetHack* behaves.

Setting the options

Options may be set in a number of ways. Within the game, the '`O`' command allows you to view all options and change most of them. You can also set options automatically by placing them in the "NETHACKOPTIONS" environment variable or in a configuration file. Some versions of *NetHack* also have front-end programs that allow you to set options before starting the game.

Using the NETHACKOPTIONS environment variable

The NETHACKOPTIONS variable is a comma-separated list of initial values for the various options. Some can only be turned on or off. You turn one of these on by adding the name of the option to the list, and turn it off by typing a '`!`' or "`no`" before the name. Others take a character string as a value. You can set string options by typing the option name, a colon or equals sign, and then the value of the string. The value is terminated by the next comma or the end of string.

For example, to set up an environment variable so that *autoquiver* is on, *autopickup* is off, the *name* is set to "Blue Meanie", and the *fruit* is set to "papaya", you would enter the command

```
setenv NETHACKOPTIONS "autoquiver,\!autopickup,name:Blue Meanie,fruit:papaya"
```

in *csh* (note the need to escape the ! since it's special to the shell), or

```
NETHACKOPTIONS="autoquiver,!autopickup,name:Blue Meanie,fruit:papaya"
export NETHACKOPTIONS
```

in *sh* or *ksh*.

Using a configuration file

Any line in the configuration file starting with "`OPTIONS=`" may be filled out with options in the same syntax as in NETHACKOPTIONS. Any line starting with "`DUNGEON=`", "`EFFECTS=`", "`MONSTERS=`", "`OBJECTS=`", "`TRAPS=`", or "`BOULDER=`" is taken as defining the corresponding *dungeon*, *effects*, *monsters*, *objects*, *traps* or *boulder* option in a different syntax, a sequence of decimal numbers giving the character position in the current font to be used in displaying each entry. A zero in any entry in such a sequence leaves the display of that entry unchanged; this feature is not available using the option syntax. Such a sequence can be continued to multiple lines by putting a '\' at the end of each line to be continued. Any line starting with '`#`' is treated as a comment.

The default name of the configuration file varies on different operating systems, but NETHACKOPTIONS can also be set to the full name of a file you want to use (possibly preceded by an '`@`').

Customization options

Here are explanations of what the various options do. Character strings that are too long may be truncated. Some of the options listed may be inactive in your dungeon.

align — Your starting alignment (`align:lawful`, `align:neutral`, or `align:chaotic`). You may specify just the first letter. The default is to randomly pick an appropriate alignment. Cannot be set with the '`O`' command.

autodig — Automatically dig if you are wielding a digging tool and moving into a place that can be dug (default false).

autopickup — Automatically pick up things onto which you move (default on).

autoquiver — This option controls what happens when you attempt the 'f' (fire) command with an empty quiver. When true, the computer will fill your quiver with some suitable weapon. Note that it will not take into account the blessed/cursed status, enchantment, damage, or quality of the weapon; you are free to manually fill your quiver with the 'Q' command instead. If no weapon is found or the option is false, the 't' (throw) command is executed instead. (default false)

boulder — Set the character used to display boulders (default is rock class symbol).

catname — Name your starting cat (ex. "`catname:Morris`"). Cannot be set with the '`O`' command.

character — Pick your type of character (ex. "`character:Monk`"); synonym for "*role*". See "*name*" for an alternate method of specifying your role. Normally only the first letter of the value is examined; the string "`random`" is an exception.

checkpoint — Save game state after each level change, for possible recovery after program crash (default on).

checkspace — Check free disk space before writing files to disk (default on). You may have to turn this off if you have more than 2 GB free space on the partition used for your save and level files. Only applies when MFLOPPY was defined during compilation.

cmdassist — Have the game provide some additional command assistance for new players if it detects some anticipated mistakes (default on).

confirm — Have user confirm attacks on pets, shopkeepers, and other peaceable creatures (default on).

DECgraphics — Use a predefined selection of characters from the DEC VT-xxx/DEC Rainbow/ANSI line-drawing character set to display the dungeon/effects/traps instead of having to define a full graphics set yourself (default off). This option also sets up proper handling of graphics characters for such terminals, so you should specify it when appropriate even if you override the selections with your own graphics strings.

disclose — Controls options for disclosing various information when the game ends (defaults to all possibilities being disclosed). The possibilities are:
`i` — disclose your inventory.
`a` — disclose your attributes.
`v` — summarize monsters that have been vanquished.
`g` — list monster species that have been genocided.
`c` — display your conduct.
Each disclosure possibility can optionally be preceded by a prefix which let you refine how it behaves. Here are the valid prefixes:
`y` — prompt you and default to yes on the prompt.
`n` — prompt you and default to no on the prompt.
`+` — disclose it without prompting.
`-` — do not disclose it and do not prompt.
(ex. "`disclose:yi na +v -g -c`") The example sets *inventory* to *prompt* and default to *yes*, *attributes* to *prompt* and default to *no*, *vanquished* to *disclose without prompting*, *genocided* to *not disclose* and not to *prompt*, and *conduct* to *not disclose* and not to *prompt*. Note that the vanquished monsters list includes all monsters

killed by traps and each other as well as by you.

dogname
Name your starting dog (ex. “`dogname:Fang`”). Cannot be set with the ‘`O`’ command.

dungeon
Set the graphics symbols for displaying the dungeon (default “` |--------||.-|++## .##<>_|\\#{}.}..## #}`”). The *dungeon* option should be followed by a string of 1–41 characters to be used instead of the default map-drawing characters. The dungeon map will use the characters you specify instead of the default symbols, and default symbols for any you do not specify. Remember that you may need to escape some of these characters on a command line if they are special to your shell.
Note that *NetHack* escape-processes this option string in conventional C fashion. This means that ‘\’ is a prefix to take the following character literally. Thus ‘\’ needs to be represented as ‘\\’. The special escape form ‘`\m`’ switches on the meta bit in the following character, and the ‘`^`’ prefix causes the following character to be treated as a control character.
The order of the symbols is: solid rock, vertical wall, horizontal wall, upper left corner, upper right corner, lower left corner, lower right corner, cross wall, upward T wall, downward T wall, leftward T wall, rightward T wall, no door, vertical open door, horizontal open door, vertical closed door, horizontal closed door, iron bars, tree, floor of a room, dark corridor, lit corridor, stairs up, stairs down, ladder up, ladder down, altar, grave, throne, kitchen sink, fountain, pool or moat, ice, lava, vertical lowered drawbridge, horizontal lowered drawbridge, vertical raised drawbridge, horizontal raised drawbridge, air, cloud, under water.
You might want to use ‘`+`’ for the corners and T walls for a more aesthetic, boxier display. Note that in the next release, new symbols may be added, or the present ones rearranged.
Cannot be set with the ‘`O`’ command.

effects
Set the graphics symbols for displaying special effects (default “`|-\\/*!)(0#@*/-\\ ||\\-//-\\| |\\-/`”). The *effects* option should be followed by a string of 1–29 characters to be used instead of the default special-effects characters. This string is subjected to the same processing as the *dungeon* option.
The order of the symbols is: vertical beam, horizontal beam, left slant, right slant, digging beam, camera flash beam, left boomerang, right boomerang, four glyphs giving the sequence for magic resistance displays, the eight surrounding glyphs for swallowed display, nine glyphs for explosions. An explosion consists of three rows (top, middle, and bottom) of three characters. The explosion is centered in the center of this 3×3 array.
Note that in the next release, new symbols may be added, or the present ones rearranged.
Cannot be set with the ‘`O`’ command.

extmenu
Changes the extended commands interface to pop-up a menu of available commands. It is keystroke compatible with the traditional interface except that it does not require that you hit Enter. It is implemented only by the tty port (default off), when the game has been compiled to support tty graphics.

female
An obsolete synonym for “`gender:female`”. Cannot be set with the ‘`O`’ command.

fixinv
An object’s inventory letter sticks to it when it’s dropped (default on). If this is off, dropping an object shifts all the remaining inventory letters.

fruit
Name a fruit after something you enjoy eating (ex. “`fruit:mango`”) (default “`slime mold`”). Basically a nostalgic whimsy that *NetHack* uses from time to time. You should set this to something you find more appetizing than slime mold. Apples, oranges, pears, bananas, and melons already exist in *NetHack*, so don’t use those.

gender
Your starting gender (`gender:male` or `gender:female`). You may specify just the

first letter. Although you can still denote your gender using the "`male`" and "`female`" options, the "`gender`" option will take precedence. The default is to randomly pick an appropriate gender. Cannot be set with the '`O`' command.

help If more information is available for an object looked at with the '`/`' command, ask if you want to see it (default on). Turning help off makes just looking at things faster, since you aren't interrupted with the "`More info?`" prompt, but it also means that you might miss some interesting and/or important information.

horsename Name your starting horse (ex. "`horsename:Trigger`"). Cannot be set with the '`O`' command.

IBMgraphics Use a predefined selection of IBM extended ASCII characters to display the dungeon/effects/traps instead of having to define a full graphics set yourself (default off). This option also sets up proper handling of graphics characters for such terminals, so you should specify it when appropriate even if you override the selections with your own graphics strings.

ignintr Ignore interrupt signals, including breaks (default off).

legacy Display an introductory message when starting the game (default on).

lit_corridor Show corridor squares seen by night vision or a light source held by your character as lit (default off).

lootabc Use the old '`a`', '`b`', and '`c`' keyboard shortcuts when looting, rather than the mnemonics '`o`', '`i`', and '`b`' (default off).

mail Enable mail delivery during the game (default on).

male An obsolete synonym for "`gender:male`". Cannot be set with the '`O`' command.

menustyle Controls the interface used when you need to choose various objects (in response to the Drop command, for instance). The value specified should be the first letter of one of the following: traditional, combination, partial, or full. Traditional was the only interface available for earlier versions; it consists of a prompt for object class characters, followed by an object-by-object prompt for all items matching the selected object class(es). Combination starts with a prompt for object class(es) of interest, but then displays a menu of matching objects rather than prompting one-by-one. Partial skips the object class filtering and immediately displays a menu of all objects. Full displays a menu of object classes rather than a character prompt, and then a menu of matching objects for selection.

menu_deselect_all Menu character accelerator to deselect all items in a menu. Implemented by the Amiga, Gem, X11 and tty ports. Default '-'.

menu_deselect_page Menu character accelerator deselect all items on this page of a menu. Implemented by the Amiga, Gem and tty ports. Default '\'.

menu_first_page Menu character accelerator to jump to the first page in a menu. Implemented by the Amiga, Gem and tty ports. Default '^'.

menu_invert_all Menu character accelerator to invert all items in a menu. Implemented by the Amiga, Gem, X11 and tty ports. Default '@'.

menu_invert_page Menu character accelerator to invert all items on this page of a menu. Implemented by the Amiga, Gem and tty ports. Default '~'.

menu_last_page Menu character accelerator to jump to the last page in a menu. Implemented by the Amiga, Gem and tty ports. Default '`|`'.

menu_next_page Menu character accelerator to goto the next menu page. Implemented by the Amiga, Gem and tty ports. Default '`>`'.

menu_previous_page Menu character accelerator to goto the previous menu page. Implemented by the Amiga, Gem and tty ports. Default '`<`'.

menu_search Menu character accelerator to search for a menu item. Implemented by the Amiga, Gem and X11 ports. Default ‘:’.

menu_select_all Menu character accelerator to select all items in a menu. Implemented by the Amiga, Gem, X11 and tty ports. Default ‘.’.

menu_select_page Menu character accelerator to select all items on this page of a menu. Implemented by the Amiga, Gem and tty ports. Default ‘,’.

monsters Set the characters used to display monster classes (default “`abcdefghijklmnopqrstuv wxyzABCDEFGHIJKLMNOPQRSTUVWXYZ@ '&;:~]`”). This string is subjected to the same processing as the *dungeon* option. The order of the symbols is ant or other insect, blob, cockatrice, dog or other canine, eye or sphere, feline, gremlin, humanoid, imp or minor demon, jelly, kobold, leprechaun, mimic, nymph, orc, piercer, quadruped, rodent, arachnid or centipede, trapper or lurker above, horse or unicorn, vortex, worm, xan or other mythical/fantastic insect, light, zruty, angelic being, bat or bird, centaur, dragon, elemental, fungus or mold, gnome, giant humanoid, invisible monster, jabberwock, Keystone Kop, lich, mummy, naga, ogre, pudding or ooze, quantum mechanic, rust monster, snake, troll, umber hulk, vampire, wraith, xorn, apelike creature, zombie, human, ghost, golem, demon, sea monster, lizard, long worm tail, and mimic. Cannot be set with the ‘`O`’ command.

msghistory The number of top line messages to save (and recall with ‘`^P`’) (default 20). Cannot be set with the ‘`O`’ command.

msg_window Allows you to change the way recalled messages are displayed. (It is currently implemented for tty only.) The possible values are:
`s` — single message (default, this was the behavior before 3.4.0).
`c` — combination, two messages as *single*, then as *full*.
`f` — full window, oldest message first.
`r` — full window, newest message first.
For backward compatibility, no value needs to be specified (which defaults to *full*), or it can be negated (which defaults to *single*).

name Set your character’s name (defaults to your user name). You can also set your character’s role by appending a dash and one or more letters of the role (that is, by suffixing one of “`-A -B -C -H -K -M -P -Ra -Ro -S -T -V -W`”). If “`-@`” is used for the role, then a random one will be automatically chosen. Cannot be set with the ‘`O`’ command.

news Read the *NetHack* news file, if present (default on). Since the news is shown at the beginning of the game, there’s no point in setting this with the ‘`O`’ command.

null Send padding nulls to the terminal (default off).

number_pad Use the number keys to move instead of `[yuhjklbn]` (default 0 or off). (number_pad:2 invokes the old DOS behaviour where ‘`5`’ means ‘`g`’, meta-‘`5`’ means ‘`G`’, and meta-‘`0`’ means ‘`I`’.)

objects Set the characters used to display object classes (default “`])[="(%!?+/$*`0_.`”). This string is subjected to the same processing as the *dungeon* option. The order of the symbols is illegal-object (should never be seen), weapon, armor, ring, amulet, tool, food, potion, scroll, spellbook, wand, gold, gem or rock, boulder or statue, iron ball, chain, and venom. Cannot be set with the ‘`O`’ command.

packorder Specify the order to list object types in (default “`")[%?+!=/(*`0_`”). The value of this option should be a string containing the symbols for the various object types. Any omitted types are filled in at the end from the previous order.

perm_invent If true, always display your current inventory in a window. This only makes sense for windowing system interfaces that implement this feature.

pettype Specify the type of your initial pet, if you are playing a character class that uses multiple types of pets; or choose to have no initial pet at all. Possible values are "cat", "dog" and "none". Cannot be set with the 'O' command.

pickup_burden When you pick up an item that would exceed this encumbrance level (Unburdened, Burdened, streSsed, straiNed, overTaxed, or overLoaded), you will be asked if you want to continue. (Default 'S').

pickup_types Specify the object types to be picked up when *autopickup* is on. Default is all types.

prayconfirm Prompt for confirmation before praying (default on).

pushweapon Using the 'w' (wield) command when already wielding something pushes the old item into your alternate weapon slot (default off).

race Selects your race (for example, "race:human"). Default is random. Cannot be set with the 'O' command.

rest_on_space Make the space bar a synonym for the '.' (rest) command (default off).

role Pick your type of character (ex. "role:Samurai"); synonym for "*character*". See "*name*" for an alternate method of specifying your role. Normally only the first letter of the value is examined; 'r' is an exception with "Rogue", Ranger", and "random" values.

runmode Controls the amount of screen updating for the map window when engaged in multi-turn movement (running via shift+direction or control+direction and so forth, or via the travel command or mouse click). The possible values are:
teleport — update the map after movement has finished;
run — update the map after every seven or so steps;
walk — update the map after each step;
crawl — like *walk*, but pause briefly after each step.
This option only affects the game's screen display, not the actual results of moving. The default is *run*; versions prior to 3.4.1 used *teleport* only. Whether or not the effect is noticeable will depend upon the window port used or on the type of terminal.

safe_pet Prevent you from (knowingly) attacking your pets (default on).

scores Control what parts of the score list you are shown at the end (ex. "scores:5top scores/4around my score/own scores"). Only the first letter of each category ('t', 'a' or 'o') is necessary.

showexp Show your accumulated experience points on bottom line (default off).

showrace Display yourself as the glyph for your race, rather than the glyph for your role (default off). Note that this setting affects only the appearance of the display, not the way the game treats you.

showscore Show your approximate accumulated score on bottom line (default off).

silent Suppress terminal beeps (default on).

sortpack Sort the pack contents by type when displaying inventory (default on).

sound Enable messages about what your character hears (default on). Note that this has nothing to do with your computer's audio capabilities. This option is only partly under player control. The game toggles it off and on during and after sleep, for example.

standout Boldface monsters and "--More--" (default off).

sparkle Display a sparkly effect when a monster (including yourself) is hit by an attack to which it is resistant (default on).

suppress_alert This option may be set to a NetHack version level to suppress alert notification messages about feature changes for that and prior versions (ex. "suppress_alert:3.3.1") *time* Show the elapsed game time in turns on bottom line (default off).

timed_delay When pausing momentarily for display effect, such as with explosions and moving objects, use a timer rather than sending extra characters to the screen. (Applies to "tty" interface only; "X11" interface always uses a timer based delay. The default is on if configured into the program.)

tombstone Draw a tombstone graphic upon your death (default on).

toptenwin Put the ending display in a NetHack window instead of on stdout (default off). Setting this option makes the score list visible when a windowing version of NetHack is started without a parent window, but it no longer leaves the score list around after game end on a terminal or emulating window.

traps Set the graphics symbols for displaying traps (default "`^^^^^^^^^^^^^^^^^"^^^^`"). The *traps* option should be followed by a string of 1–22 characters to be used instead of the default traps characters. This string is subjected to the same processing as the *dungeon* option.
The order of the symbols is: arrow trap, dart trap, falling rock trap, squeaky board, bear trap, land mine, rolling boulder trap, sleeping gas trap, rust trap, fire trap, pit, spiked pit, hole, trap door, teleportation trap, level teleporter, magic portal, web, statue trap, magic trap, anti-magic field, polymorph trap.
Cannot be set with the '`O`' command.

travel Allow the travel command (default on). Turning this option off will prevent the game from attempting unintended moves if you make inadvertent mouse clicks on the map window.

verbose Provide more commentary during the game (default on).

windowtype Select which windowing system to use, such as "`tty`" or "`X11`" (default depends on version). Cannot be set with the '`O`' command.

Window Port Customization options

Here are explanations of the various options that are used to customize and change the characteristics of the windowtype that you have chosen. Character strings that are too long may be truncated. Not all window ports will adjust for all settings listed here. You can safely add any of these options to your config file, and if the window port is capable of adjusting to suit your preferences, it will attempt to do so. If it can't it will silently ignore it. You can find out if an option is supported by the window port that you are currently using by checking to see if it shows up in the Options list. Some options are dynamic and can be specified during the game with the '`O`' command.

align_message Where to align or place the message window (top, bottom, left, or right)

align_status Where to align or place the status window (top, bottom, left, or right).

ascii_map NetHack should display an ascii map if it can.

color NetHack should display color if it can for different monsters, objects, and dungeon features

eight_bit_tty Pass eight-bit character values (for example, specified with the *traps* option) straight through to your terminal (default off).

font_map NetHack should use a font by the chosen name for the map window.

font_menu NetHack should use a font by the chosen name for menu windows.

font_message NetHack should use a font by the chosen name for the message window.

font_status NetHack should use a font by the chosen name for the status window.

font_text NetHack should use a font by the chosen name for text windows.

font_size_map NetHack should use this size font for the map window.

font_size_menu NetHack should use this size font for menu windows.

font_size_message NetHack should use this size font for the message window.

font_size_status NetHack should use this size font for the status window.

font_size_text NetHack should use this size font for text windows.

fullscreen NetHack should try and display on the entire screen rather than in a window.

hilite_pet Visually distinguish pets from similar animals (default off). The behavior of this option depends on the type of windowing you use. In text windowing, text highlighting or inverse video is often used; with tiles, generally displays a heart symbol near pets.

large_font NetHack should use a large font.

map_mode NetHack should display the map in the manner specified.

mouse_support Allow use of the mouse for input and travel.

player_selection NetHack should pop up dialog boxes or use prompts for character selection.

popup_dialog NetHack should pop up dialog boxes for input.

preload_tiles NetHack should preload tiles into memory. For example, in the protected mode MSDOS version, control whether tiles get pre-loaded into RAM at the start of the game. Doing so enhances performance of the tile graphics, but uses more memory. (default on). Cannot be set with the 'O' command.

scroll_amount NetHack should scroll the display by this number of cells when the hero reaches the scroll_margin.

scroll_margin NetHack should scroll the display when the hero or cursor is this number of cells away from the edge of the window.

softkeyboard Display an onscreen keyboard. Handhelds are most likely to support this option.

splash_screen NetHack should display an opening splash screen when it starts up (default yes).

tiled_map NetHack should display a tiled map if it can.

tile_file Specify the name of an alternative tile file to override the default.

tile_height Specify the preferred height of each tile in a tile capable port.

tile_width Specify the preferred width of each tile in a tile capable port

use_inverse NetHack should display inverse when the game specifies it.

vary_msgcount NetHack should display this number of messages at a time in the message window.

windowcolors NetHack should display windows with the specified foreground/background colors if it can.

wraptext NetHack port should wrap long lines of text if they don't fit in the visible area of the window.

Platform-specific Customization options

Here are explanations of options that are used by specific platforms or ports to customize and change the port behaviour.

altkeyhandler Select an alternate keystroke handler dll to load (*Win32 tty NetHack* only). The name of the handler is specified without the .dll extension and without any path information. Cannot be set with the 'O' command.

altmeta (default on, *Amiga NetHack* only).

BIOS Use BIOS calls to update the screen display quickly and to read the keyboard (allowing the use of arrow keys to move) on machines with an IBM PC compatible BIOS ROM (default off, *OS/2, PC* and *ST NetHack* only).

flush (default off, *Amiga NetHack* only).

Macgraphics (default on, *Mac NetHack* only).

page_wait (default off, *Mac NetHack* only).

rawio Force raw (non-cbreak) mode for faster output and more bulletproof input (MS-DOS sometimes treats '`^P`' as a printer toggle without it) (default off, *OS/2, PC* and *ST NetHack* only). Note: DEC Rainbows hang if this is turned on. Cannot be set with the '`O`' command.

soundcard (default off, *PC NetHack* only). Cannot be set with the '`O`' command.

subkeyval (*Win32 tty NetHack* only). May be used to alter the value of keystrokes that the operating system returns to NetHack to help compensate for international keyboard issues. OPTIONS=subkeyvalue:171/92 will return 92 to NetHack, if 171 was originally going to be returned. You can use multiple subkeyvalue statements in the config file if needed. Cannot be set with the '`O`' command.

video Set the video mode used (*PC NetHack* only). Values are *autodetect*, *default*, or *vga*. Setting *vga* (or *autodetect* with vga hardware present) will cause the game to display tiles. Cannot be set with the '`O`' command.

videocolors Set the color palette for PC systems using NO_TERMS (default 4-2-6-1-5-3-15-12-10-14-9-13-11, *PC NetHack* only). The order of colors is red, green, brown, blue, magenta, cyan, bright.white, bright.red, bright.green, yellow, bright.blue, bright.magenta, and bright.cyan. Cannot be set with the '`O`' command.

videoshades Set the intensity level of the three gray scales available (default dark normal light, *PC NetHack* only). If the game display is difficult to read, try adjusting these scales; if this does not correct the problem, try `!color`. Cannot be set with the '`O`' command.

Configuring User Sounds

Some platforms allow you to define sound files to be played when a message that matches a user-defined pattern is delivered to the message window. At this time the Qt port and the win32tty and win32gui ports support the use of user sounds.

The following config file entries are relevant to mapping user sounds to messages:

SOUNDDIR The directory that houses the sound files to be played.

SOUND An entry that maps a sound file to a user-specified message pattern. Each SOUND entry is broken down into the following parts:
`MESG` — message window mapping (the only one supported in 3.4).
`pattern` — the pattern to match.
`sound file` — the sound file to play.
`volume` — the volume to be set while playing the sound file.

The exact format for the pattern depends on whether the platform is built to use *regular expressions* or NetHack's own internal pattern matching facility. The *regular expressions* matching can be much more sophisticated than the internal NetHack pattern matching, but requires 3rd party libraries on some platforms. There are plenty of references available elsewhere for explaining *regular expressions* . You can verify which pattern matching is used by your port with the #version command.

NetHack's internal pattern matching routine uses the following special characters in its pattern matching:

```
*--- matches 0 or more characters.
?--- matches any single character.
```

Here's an example of a sound mapping using NetHack's internal pattern matching facility:

```
SOUND=MESG "*chime of a cash register*" "gong.wav" 50
```

specifies that any message with "chime of a cash register" contained in it will trigger the playing of "gong.wav". You can have multiple SOUND entries in your config file.

Configuring NetHack for Play by the Blind

NetHack can be set up to use only standard ASCII characters for making maps of the dungeons. This makes the MS-DOS versions of NetHack completely accessible to the blind who use speech and/or Braille access technologies. Players will require a good working knowledge of their screen-reader's review features, and will have to know how to navigate horizontally and vertically character by character. They will also find the search capabilities of their screen-readers to be quite valuable. Be certain to examine this Guidebook before playing so you have an idea what the screen layout is like. You'll also need to be able to locate the PC cursor. It is always where your character is located. Merely searching for an @-sign will not always find your character since there are other humanoids represented by the same sign. Your screen-reader should also have a function which gives you the row and column of your review cursor and the PC cursor. These co-ordinates are often useful in giving players a better sense of the overall location of items on the screen. While it is not difficult for experienced users to edit the *defaults.nh* file to accomplish this, novices may find this task somewhat daunting. Included in all official distributions of NetHack is a file called *NHAccess.nh*. Replacing *defaults.nh* with this file will cause the game to run in a manner accessible to the blind. After you have gained some experience with the game and with editing files, you may want to alter settings to better suit your preferences. Instructions on how to do this are included in the *NHAccess.nh* file itself. The most crucial settings to make the game accessible are:

IBMgraphics Disable IBMgraphics by commenting out this option.

menustyle:traditional This will assist in the interface to speech synthesizers.

number_pad A lot of speech access programs use the number-pad to review the screen. If this is the case, disable the number_pad option and use the traditional Rogue-like commands.

Character graphics Comment out all character graphics sets found near the bottom of the *defaults.nh* file. Most of these replace *NetHack*'s default representation of the dungeon using standard ASCII characters with fancier characters from extended character sets, and these fancier characters can annoy screen-readers.

10 Scoring

NetHack maintains a list of the top scores or scorers on your machine, depending on how it is set up. In the latter case, each account on the machine can post only one non-winning score on this list. If you score higher than someone else on this list, or better your previous score, you will be inserted in the proper place under your current name. How many scores are kept can also be set up when *NetHack* is compiled.

Your score is chiefly based upon how much experience you gained, how much loot you accumulated, how deep you explored, and how the game ended. If you quit the game, you escape with all of your gold intact. If, however, you get killed in the Mazes of Menace, the guild will only hear about 90 % of your gold when your corpse is discovered (adventurers have been known to collect finder's fees). So, consider whether you want to take one last hit at that monster and possibly live, or quit and stop with whatever you have. If you quit, you keep all your gold, but if you swing and live, you might find more.

If you just want to see what the current top players/games list is, you can type

```
nethack -s all
```

on most versions.

11 Explore mode

NetHack is an intricate and difficult game. Novices might falter in fear, aware of their ignorance of the means to survive. Well, fear not. Your dungeon may come equipped with an "explore" or "discovery" mode that enables you to keep old save files and cheat death, at the paltry cost of not getting on the high score list.

There are two ways of enabling explore mode. One is to start the game with the `-X` switch. The other is to issue the '`X`' command while already playing the game. The other benefits of explore mode are left for the trepid reader to discover.

12 Credits

The original *hack* game was modeled on the Berkeley UNIX *rogue* game. Large portions of this paper were shamelessly cribbed from *A Guide to the Dungeons of Doom*, by Michael C. Toy and Kenneth C. R. C. Arnold. Small portions were adapted from *Further Exploration of the Dungeons of Doom*, by Ken Arromdee.

NetHack is the product of literally dozens of people's work. Main events in the course of the game development are described below:

Jay Fenlason wrote the original *Hack* with help from *Kenny Woodland*, *Mike Thome*, and *Jon Payne*.

Andries Brouwer did a major re-write, transforming *Hack* into a very different game, and published (at least) three versions (1.0.1, 1.0.2, and 1.0.3) for UNIX machines to the Usenet.

Don G. Kneller ported *Hack* 1.0.3 to Microsoft C and MS-DOS, producing *PC Hack* 1.01e, added support for DEC Rainbow graphics in version 1.03g, and went on to produce at least four more versions (3.0, 3.2, 3.51, and 3.6).

R. Black ported *PC Hack* 3.51 to Lattice C and the Atari 520/1040ST, producing *ST Hack* 1.03.

Mike Stephenson merged these various versions back together, incorporating many of the added features, and produced *NetHack* version 1.4. He then coordinated a cast of thousands in enhancing and debugging *NetHack* 1.4 and released *NetHack* versions 2.2 and 2.3.

Later, Mike coordinated a major rewrite of the game, heading a team which included *Ken Arromdee*, *Jean-Christophe Collet*, *Steve Creps*, *Eric Hendrickson*, *Izchak Miller*, *Eric S. Raymond*, *John Rupley*, *Mike Threepoint*, and *Janet Walz*, to produce *NetHack* 3.0c.

NetHack 3.0 was ported to the Atari by *Eric R. Smith*, to OS/2 by *Timo Hakulinen*, and to VMS by *David Gentzel*. The three of them and *Kevin Darcy* later joined the main development team to produce subsequent revisions of 3.0.

Olaf Seibert ported *NetHack* 2.3 and 3.0 to the Amiga. *Norm Meluch*, *Stephen Spackman* and *Pierre Martineau* designed overlay code for *PC NetHack* 3.0. *Johnny Lee* ported *NetHack* 3.0 to the Macintosh. Along with various other Dungeoneers, they continued to enhance the PC, Macintosh, and Amiga ports through the later revisions of 3.0.

Headed by *Mike Stephenson* and coordinated by *Izchak Miller* and *Janet Walz*, the development team which now included *Ken Arromdee*, *David Cohrs*, *Jean-Christophe Collet*, *Kevin Darcy*, *Matt Day*, *Timo Hakulinen*, *Steve Linhart*, *Dean Luick*, *Pat Rankin*, *Eric Raymond*, and *Eric Smith* undertook a radical revision of 3.0. They re-structured the game's design, and re-wrote major parts of the code. They added multiple dungeons, a new display, special individual character quests, a new endgame and many other new features, and produced *NetHack* 3.1.

Ken Lorber, *Gregg Wonderly* and *Greg Olson*, with help from *Richard Addison*, *Mike Passaretti*, and *Olaf Seibert*, developed *NetHack* 3.1 for the Amiga.

Norm Meluch and *Kevin Smolkowski*, with help from *Carl Schelin*, *Stephen Spackman*, *Steve VanDevender*, and *Paul Winner*, ported *NetHack* 3.1 to the PC.

Jon W{tte and *Hao-yang Wang*, with help from *Ross Brown*, *Mike Engber*, *David Hairston*, *Michael Hamel*, *Jonathan Handler*, *Johnny Lee*, *Tim Lennan*, *Rob Menke*, and *Andy Swanson*, developed *NetHack* 3.1 for the Macintosh, porting it for MPW. Building on their development, *Barton House* added a Think C port.

Timo Hakulinen ported *NetHack* 3.1 to OS/2. *Eric Smith* ported *NetHack* 3.1 to the Atari. *Pat Rankin*, with help from *Joshua Delahunty*, was responsible for the VMS version of *NetHack* 3.1. *Michael Allison* ported *NetHack* 3.1 to Windows NT.

Dean Luick, with help from *David Cohrs*, developed *NetHack* 3.1 for X11. *Warwick Allison* wrote a tiled version of NetHack for the Atari; he later contributed the tiles to the DevTeam and tile support was then added to other platforms.

The 3.2 development team, comprised of *Michael Allison*, *Ken Arromdee*, *David Cohrs*, *Jessie Collet*, *Steve Creps*, *Kevin Darcy*, *Timo Hakulinen*, *Steve Linhart*, *Dean Luick*, *Pat Rankin*, *Eric Smith*, *Mike Stephenson*, *Janet Walz*, and *Paul Winner*, released version 3.2 in April of 1996.

Version 3.2 marked the tenth anniversary of the formation of the development team. In a testament to their dedication to the game, all thirteen members of the original development team remained on the team at the start of work on that release. During the interval between the release of 3.1.3 and 3.2, one of the founding members of the development team, *Dr. Izchak Miller*, was diagnosed with cancer and passed away. That release of the game was dedicated to him by the development and porting teams.

During the lifespan of *NetHack* 3.1 and 3.2, several enthusiasts of the game added their own modifications to the game and made these "variants" publicly available:

Tom Proudfoot and *Yuval Oren* created *NetHack++*, which was quickly renamed *NetHack−−*. Working independently, *Stephen White* wrote *NetHack Plus*. *Tom Proudfoot* later merged *NetHack Plus* and his own *NetHack−−* to produce *SLASH*. *Larry Stewart-Zerba* and *Warwick Allison* improved the spell casting system with the Wizard Patch. *Warwick Allison* also ported NetHack to use the Qt interface.

Warren Cheung combined *SLASH* with the Wizard Patch to produce *Slash'em*, and with the help of *Kevin Hugo*, added more features. Kevin later joined the DevTeam and incorporated the best of these ideas into NetHack 3.3.

The final update to 3.2 was the bug fix release 3.2.3, which was released simultaneously with 3.3.0 in December 1999 just in time for the Year 2000.

The 3.3 development team, consisting of *Michael Allison*, *Ken Arromdee*, *David Cohrs*, *Jessie Collet*, *Steve Creps*, *Kevin Darcy*, *Timo Hakulinen*, *Kevin Hugo*, *Steve Linhart*, *Ken Lorber*, *Dean Luick*, *Pat Rankin*, *Eric Smith*, *Mike Stephenson*, *Janet Walz*, and *Paul Winner*, released 3.3.0 in December 1999 and 3.3.1 in August of 2000.

Version 3.3 offered many firsts. It was the first version to separate race and profession. The Elf class was removed in preference to an elf race, and the races of dwarves, gnomes, and orcs made their first appearance in the game alongside the familiar human race. Monk and Ranger roles joined Archeologists, Barbarians, Cavemen, Healers, Knights, Priests, Rogues, Samurai, Tourists, Valkyries and of course, Wizards. It was also the first version to allow you to ride a steed, and was the first version to have a publicly available web-site listing all the bugs that had been discovered. Despite that constantly growing bug list, 3.3 proved stable enough to last for more than a year and a half.

The 3.4 development team initially consisted of *Michael Allison*, *Ken Arromdee*, *David Cohrs*, *Jessie Collet*, *Kevin Hugo*, *Ken Lorber*, *Dean Luick*, *Pat Rankin*, *Mike Stephenson*, *Janet Walz*, and *Paul Winner*, with *Warwick Allison* joining just before the release of NetHack 3.4.0 in March 2002.

As with version 3.3, various people contributed to the game as a whole as well as supporting ports on the different platforms that *NetHack* runs on:

Pat Rankin maintained 3.4 for VMS.

Michael Allison maintained NetHack 3.4 for the MS-DOS platform. *Paul Winner* and *Yitzhak Sapir* provided encouragement.

Dean Luick, *Mark Modrall*, and *Kevin Hugo* maintained and enhanced the Macintosh port of 3.4.

Michael Allison, *David Cohrs*, *Alex Kompel*, *Dion Nicolaas*, and *Yitzhak Sapir* maintained and enhanced 3.4 for the Microsoft Windows platform. *Alex Kompel* contributed a new graphical interface for the Windows port. *Alex Kompel* also contributed a Windows CE port for 3.4.1.

Ron Van Iwaarden maintained 3.4 for OS/2.

Janne Salmijärvi and *Teemu Suikki* maintained and enhanced the Amiga port of 3.4 after *Janne Salmijärvi* resurrected it for 3.3.1.

Christian "Marvin" Bressler maintained 3.4 for the Atari after he resurrected it for 3.3.1.

There is a NetHack web site maintained by *Ken Lorber* at http://www.nethack.org/.

Dungeoneers

From time to time, some depraved individual out there in netland sends a particularly intriguing modification to help out with the game. The Gods of the Dungeon sometimes make note of the names of the worst of these miscreants in this, the list of Dungeoneers:

Adam Aronow
Alex Kompel
Andreas Dorn
Andy Church
Andy Swanson
Ari Huttunen
Barton House
Benson I. Margulies
Bill Dyer
Boudewijn Waijers
Bruce Cox
Bruce Holloway
Bruce Mewborne
Carl Schelin
Chris Russo
David Cohrs
David Damerell
David Gentzel
David Hairston
Dean Luick
Del Lamb
Deron Meranda
Dion Nicolaas
Dylan O'Donnell
Eric Backus
Eric Hendrickson
Eric R. Smith
Eric S. Raymond
Erik Andersen
Frederick Roeber
Gil Neiger
Greg Laskin
Greg Olson
Gregg Wonderly
Hao-yang Wang
Helge Hafting
Irina Rempt-Drijfhout
Izchak Miller
J. Ali Harlow
Janet Walz
Janne Salmijärvi
Jean-Christophe Collet
Jochen Erwied
John Kallen
John Rupley
John S. Bien
Johnny Lee
Jon W{tte
Jonathan Handler
Joshua Delahunty
Keizo Yamamoto
Ken Arnold
Ken Arromdee
Ken Lorber
Ken Washikita
Kevin Darcy
Kevin Hugo
Kevin Sitze
Kevin Smolkowski
Kevin Sweet
Lars Huttar
Malcolm Ryan
Mark Gooderum
Mark Modrall
Marvin Bressler
Matthew Day
Merlyn LeRoy
Michael Allison
Michael Feir
Michael Hamel
Michael Sokolov
Mike Engber
Mike Gallop
Mike Passaretti
Mike Stephenson
Norm Meluch
Olaf Seibert
Pasi Kallinen
Pat Rankin
Paul Winner
Pierre Martineau
Ralf Brown
Ray Chason
Richard Addison
Richard Beigel
Richard P. Hughey
Rob Menke
Robin Johnson
Roderick Schertler
Roland McGrath
Ron Van Iwaarden
Ronnen Miller
Ross Brown
Sascha Wostmann
Scott Bigham
Scott R. Turner
Stephen Spackman
Stephen White
Steve Creps
Steve Linhart
Steve VanDevender
Teemu Suikki
Tim Lennan
Timo Hakulinen
Tom Almy
Tom West
Warren Cheung
Warwick Allison
Yitzhak Sapir