Stones of Chaos: True God

Blasphemy, as it turns out, wasn't the best idea you've ever had, tho it may have seemed so at the time. So with a sword of death hanging over your head you decide to do the one thing you can to save your life. You decide to switch deities and pay respects to the one force in the heavens that may or may not grant you the right to live: Arnie the Random Number God. But the RNG is notoriously unpredictable and pleasing him will involve deciphering cryptic clues as to the tasks you can perform to amuse him and improve your standing with him, all while fighting your way through a dungeon to recover the Stone of Chaos on the bottom most level and restore it to its altar on the top most level. Along the way you could be completely randomly effected by one of the RNG's completely random whims. These effects could be good or bad.

If luck is on your side you may find yourself at a point where winning becomes inevitable. If not… well you're no worse off.

SoC:tTG packs a surprising amount of complexity into a simple game. The way that it takes advantage of the small fiddling of variables that any game designer will tell you could eat up an entire development cycle as an integral part of the gameplay makes no two games the same, in their own way.

/* map.h listing begins: */

```
#ifndef MAP_H_INCLUDED
#define MAP_H_INCLUDED
#define YSIZE 21
#define XSIZE 80
#define NPOINT 30
#define NCONNECT 2
#define TUNPOINT 5
#define Y 0
#define X 1

int dlvl;
char map[YSIZE][XSIZE];
int upy,upx,downy,downx;
void genmap(int maptype);
void drawmap();
void gendungeon();
void genbasic();
void gentunnel();
void genforest();
void genempty();
void nextlevel();
void prevlevel();
bool seen[YSIZE][XSIZE];

#endif // MAP_H_INCLUDED
```

/* misc.h listing begins: */

```
#ifndef MISC_H_INCLUDED
#define MISC_H_INCLUDED
void init ();
void putmsg(char newmess[]);
double range(int y1,int x1,int y2, int x2);
int dice(int numDice,int sideDice,int bonus);
void help();

#endif // MISC_H_INCLUDED
```

/* ent.h listing begins: */

```
#ifndef ENT_H_INCLUDED
#define ENT_H_INCLUDED
#include"map.h"
#define NMON 12

bool stone=false;
int exper;

void look();
bool playerturn();
bool playerinput();
void genplayer();
void see(int y,int x);
void clearlocs();
void genmon(int i,int lv);
void monturn(int i);
void action();
void levelmonster(int i);
void demotemonster(int i);
void levelplayer();
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
void demoteplayer();


struct ent
{
int x,y,hp,mhp,spd,str,lvl,vis,counter;
char sym;
bool alive;
}ents[NMON];

void pattack(ent* attackee);
void monattack(ent* mon);

ent* entsl[YSIZE][XSIZE];

#endif // ENT_H_INCLUDED
```

/* item.h listing begins: */

```
#ifndef ITEM_H_INCLUDED
#define ITEM_H_INCLUDED
#include "map.h"
#define MAXITEM 20
int numitems=4;


struct items{
   int x,y;
   char sym;
}item[MAXITEM];

void genitem(int i);
void useitem(items* it);
void destitem(items* it);
void putitems();
items* iteml[YSIZE][XSIZE];


#endif // ITEM_H_INCLUDED
```

/* arnie.h listing begins: */

```
#ifndef ARNIE_H_INCLUDED
#define ARNIE_H_INCLUDED
#define NTHOUGHT 45
#define NWHIMS 10
#define NPLAYS 14

#define KILLTWO 0
#define KILLSEVEN 1
#define GOFAST 2
#define GOSLOW 3
#define PACIFIST 4
#define SLAUGHTER 5
#define NOITEMS 6
#define ALLITEMS 7
#define GOBACK 8
#define GETOUT 9
```

If you're having trouble deciphering the meaning of the messages at the beginning of the game this handy guide should help:

"You feel like killing only a couple people right now" - You should kill exactly 2 random souls before leaving the current level.

"7 souls will fall." - Kill exactly 7 random souls and leave the level.

"HURRY UP!!!" - Finish the current level in 100 turns or less.

"Slooooowwww doooowwwwnnn" - Don't leave the level before 200 turns are up.

"You feel like a hippy" - Kill nothing. If you succeed you will gain a level.

"You think genocide mite be a fun thing to do"- You gain a small bonus and a tiny bit of HP for each enemy you kill.

"This water tastes strange" - If you drink any potions, the amount of potions on farther levels will be halved and if not it will be doubled.

"I'm thirsty" - If you drink all the potions on this level you will receive more on further levels if you don't drink them all you will get less.

"Someone is partying behind you" - Just go back the way you came.

"DAMN" "A COP!" "RUN!!" - There is only one enemy, but his touch is instant death. Similar to the game "Get Out!!"

Along the way these

/* Listing continued on next page…*/

random events could befall you:

"You are pulled in a non-existent direction" - Teleport to a different level depending on how much the RN God likes you.

"The RN God touches you." - You are damaged or healed depending on how much Arnie likes you.

"Somebody gets stronger." - Between 1 and 5 characters on the level (you included) gain a level. (This can be deadly on "Get Out!!" levels if the RNG hates you.)

"Arnie is bored with you." - The RNG starts liking you less.

"You amuse the Random Number God" - The RNG starts liking you more.

"Barfff... ugggh" - You lose a level.

"A hot wind passes through here" - Everybody on the level takes some damage.

"Beer crosses your mind" - The amount of potions on each level is changed depending on how much the Random Number God Favors you.

"You feel a very soothing wind" - Everybody on the level receives some HP.

"Time seems to be distorted" - Your Speed changes depending on the Random Number Generator's level of pleasure towards you.

"Your eyes feel weird." - How far you can see is changed.

"The dead arise" - All dead monsters on the level come back to life.

/* Listing continued from previous page */

```
char thought[NTHOUGHT][80]=
{
"A kitten",
"A girl named Moo",
"Chewing gum and baling wire",
"Nachos",
"Nothing much, What are you thinking?",
"Commie invaders",
"Get Out",
"Buttons and such",
"Lemons",
"#rgrd on quakenet irc",
"Existance",
"Nonexistance",
"Who would win in a fight, me or kos?",
"http://cymonsgames.retroremakes.com/commieinvaders",
"Pumpkin pie spice",
"Pretty pretty princess",
"playing harmonica",
"How much wood could a woodchuck chuck if i had a beer?",
"Ice cold bile",
"Who would win in a fight, Tesla or Mr. T?",
"Playing chess with checkers pieces",
"I'm on a roll baby",
"A guy that does not exist",
"Changing his name to '><(0)|\\/|'",
"Changing his name to 'Dolly'",
"Yo Quiero Taco bell",
"Quien es la mas fea, Hilary Clinton o Brittany Spears?",
"A one-eyed picanese",
"War is bad",
"War is good",
"I wonder what it is like to be mortal",
":)",
":(",
"Should i go to KFC or kill a chicken myself?",
"Sex and candy",
"7DRL winner Whoooo!!",
"Why do people go to work all the time?",
"How many of thes stupid phrases do I really need?",
"Watching an episode of 'Doug'",
"A song to make me stop killing you",
"Margays can turn their hands backwards",
"Haydn's 94th Symphony in G",
"I wonder what you taste like",
"A wizard did it",
"If you blame anyone, blame Bunny.",
};

int wi;
bool noquestion;
bool whim[NWHIMS];
int turncount,killcount,itemcount;
int luck;
int boredom;
int question();
void newwhim();
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
void checkitems(int desired);

void checkspeed(bool desired);
void checkkills(int desired);
void getout();
int arniemod();
void bored();


#endif // ARNIE_H_INCLUDED
```

If you need help in the game press the '?' key.

Stones of Chaos: The True God is a 7 Day Roguelike written by James E. Ward. The version listed here has been modified from the original 7DRL version.

.

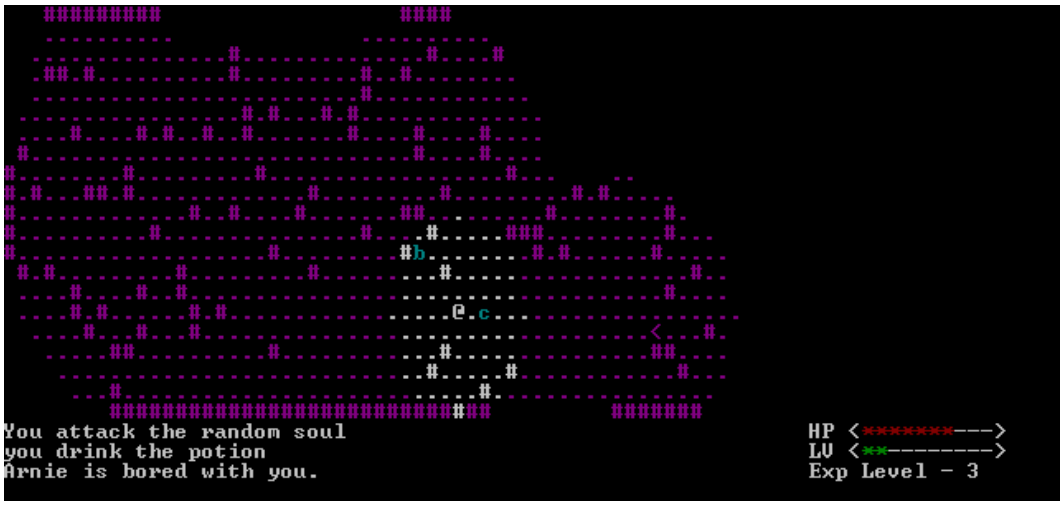
/* main.,cpp listing begins: */

```
#include <curses.h>
#include<cstdlib>
#include<time.h>
#include<string.h>
#include"map.h"
#include"misc.h"
#include"ent.h"
#include <math.h>
#include"item.h"
#include "arnie.h"


using namespace std;

int input=0;

int main() {
   boredom=dice(30,5,0);
   noquestion=false;
   init();
   clear();
   mvprintw(0,0,"Stones of Chaos: The True God 7drl"
   mvprintw(0,1,By James E. Ward(idontexist) Cymon's Games version");
   mvprintw(3,3,"You have fallen out of the favor of your god and now he wants you dead.");
   mvprintw(4,0,"Your only chance for survival is to follow the god of randomness, He has");
   mvprintw(5,0,"only one favor to ask of you. You must recover one of the Stones of chaos");
   mvprintw(6,0,"and sacrifice it on his altar.");
   getch();
   dlvl=1;
   genmap(0);
   clearlocs();
   genplayer();
   for (int i=1;i<NMON;i++) {
      genmon(i,dlvl);
   }
   putitems();
   map[ents[0].y][ents[0].x]='.';
   drawmap();
   wi=NWHIMS+1;
   for (int i=0;i<NWHIMS;i++) {
      whim[i]=false;
   }
   newwhim();
   while (1) {
      if (playerturn()) {
         for (int i=1;i<NMON;i++) {
```



/* Listing continued on next page…*/

/* Listing continued from previous page */

```
            monturn(i);
         }
      }
      drawmap();
      if (ents[0].hp<1) {
         clear();
         mvprintw(0,5,"You are dead, The RNG must not have liked you");
         mvprintw(3,0,"press space to exit");

         while (1) {
            input=getch();
            if (input==' ') {
               exit(endwin());
            }
         }
         exit(endwin());
      }
   }
   exit(endwin());
}

void look() {
   int crsy=ents[0].y, crsx=ents[0].x;
   while (1) {
      drawmap();
      if (crsy==0) {
         crsy=YSIZE-2;
      }
      if (crsy==YSIZE-1) {
         crsy=1;
      }
      if (crsx==0) {
         crsx=XSIZE-2;
      }
      if (crsx==XSIZE-1) {
         crsx=1;
      }
      mvaddch(crsy,crsx,'*');
      input=getch();
      switch (input) {
      case '1':
         crsy++;
         crsx--;
         break;
      case '2':
         crsy++;
         break;
      case '3':
         crsy++;
         crsx++;
         break;
      case '4':
         crsx--;
         break;
      case '6':
         crsx++;
         break;
      case '7':
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
            crsy--;
            crsx--;
            break;
        case '8':
            crsy--;
            break;
        case '9':
            crsy--;
            crsx++;
            break;
        case '5':
            see(crsy,crsx);
            return;
        case ';':
            see(crsy,crsx);
            return;

        }
    }
}

void see(int y,int x) {
    if (iteml[y][x]!=NULL) {
        putmsg("There is a potion here.");
    } else if (entsl[y][x]!=NULL) {
        putmsg("You see a random soul");
    } else if (map[y][x]=='_') {
        putmsg("An ancient altar is here");
    } else if (map[y][x]=='*') {
        putmsg("Its the Stone of Chaos!");
    } else if (map[y][x]=='>') {
        putmsg("You see a way to the next level");
    } else if (map[y][x]=='<') {
        putmsg("You see a way to the previous level");
    } else if (map[y][x]=='.') {
        putmsg("You see an empty floor");
    } else if (map[y][x]=='#') {
        putmsg("You see a wall");
    }
}

void genplayer() {
    ents[0].sym='@';
    ents[0].mhp=100;
    ents[0].hp=100;
    ents[0].str=10;
    ents[0].spd=11+(rand()%2);
    ents[0].lvl=1;
    ents[0].y=upy;
    ents[0].x=upx;
    ents[0].vis=4;
    entsl[ents[0].y][ents[0].x]=&ents[0];
    ents[0].counter=0;
    exper=0;
    luck=0;
}

void levelplayer() {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
   ents[0].lvl++;
   ents[0].mhp+=dice(12,2,0)+arniemod();
   ents[0].str+=dice(4,3,0)+arniemod();
   ents[0].spd+=(rand()%2)+arniemod();
   ents[0].vis+=(rand()%2)+arniemod();
   ents[0].hp+=(ents[0].mhp/2);
   putmsg("You feel experienced");
}

void demoteplayer() {
   if (ents[0].lvl==1) {
      return;
   }
   ents[0].lvl--;
   ents[0].mhp-=dice(12,2,0);
   ents[0].str-=dice(4,3,0);
   ents[0].spd-=(rand()%2);
   ents[0].vis-=(rand()%2);
   if (ents[0].hp>ents[0].mhp) {
      ents[0].hp=ents[0].mhp;
   }

}

void genmon(int i,int lv) {
   ents[i].sym='a';
   ents[i].mhp=dice(5,6,0)-arniemod();
   ents[i].hp=ents[i].mhp;
   ents[i].str=dice(4,4,0)-arniemod();
   ents[i].spd= 10-arniemod();
   ents[i].vis=dice(3,2,0)-arniemod();
   ents[i].alive=true;
   ents[i].lvl=lv+(rand()%3-1)-arniemod();
   if (ents[i].lvl<1) {
      ents[i].lvl=1;
   }
   do {
      ents[i].y=rand()%(YSIZE-2)+1;
      ents[i].x=rand()%(XSIZE-2)+1;
   } while (map[ents[i].y][ents[i].x]=='#'||entsl[ents[i].y][ents[i].x]!=NULL);
   entsl[ents[i].y][ents[i].x]=&ents[i];

   levelmonster(i);
   ents[i].counter=0;
}

void levelmonster(int i) {
   for (int j=1;j<ents[i].lvl;j++) {
      ents[i].sym++;
      ents[i].mhp+=dice(4,5,0)-arniemod();
      ents[i].str+=dice(2,2,0)-arniemod();
      ents[i].spd+=rand()%2-arniemod();
      ents[i].vis+=rand()%2-arniemod();
   }
   ents[i].hp=ents[i].mhp;
}

void demotemonster(int i) {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
    ents[i].mhp-=dice(3,3,0);
    ents[i].str-=dice(2,2,0);
    ents[i].spd-=rand()%2;
    ents[i].vis-=rand()%2;
    if (ents[i].mhp<1) {
        ents[i].mhp=1;
    }
    if (ents[i].hp>ents[i].mhp) {
        ents[i].hp=ents[i].mhp;
    }
    if (ents[i].mhp<1) {
        ents[i].mhp=1;
    }
}

void monturn(int i) {
    ents[i].counter+=ents[i].spd;
    if (ents[i].counter<=100) {
        return;
    }

    ents[i].counter-=100;
    entsl[ents[i].y][ents[i].x]=NULL;
    bool up=(ents[i].y>ents[0].y);
    bool down=(ents[i].y<ents[0].y);
    bool left=(ents[i].x>ents[0].x);
    bool right=(ents[i].x<ents[0].x);
    bool ul =(up && left);
    bool ur=(up && right);
    bool dl = (down && left);
    bool dr=(down && right);

    if (range(ents[i].y,ents[i].x,ents[0].y,ents[0].x)<ents[i].vis && ents[i].alive==true)

        if (range(ents[i].y,ents[i].x,ents[0].y,ents[0].x)<2) {
            monattack(&ents[i]);
        } else
        if (ul&&entsl[ents[i].y-1][ents[i].x-1]==NULL&&map[ents[i].y-1][ents[i].x-1]!='#') {
            ents[i].y--;
            ents[i].x--;
        } else
        if (ur&&entsl[ents[i].y-1][ents[i].x+1]==NULL&&map[ents[i].y-1][ents[i].x+1]!='#') {
            ents[i].y--;
            ents[i].x++;
        } else
        if (dl&&entsl[ents[i].y+1][ents[i].x-1]==NULL&&map[ents[i].y+1][ents[i].x-1]!='#') {
            ents[i].y++;
            ents[i].x--;
        } else
        if (dr&&entsl[ents[i].y+1][ents[i].x+1]==NULL&&map[ents[i].y+1][ents[i].x+1]!='#') {
            ents[i].y++;
            ents[i].x++;
        } else if (up&&entsl[ents[i].y-1][ents[i].x]==NULL&&map[ents[i].y-1][ents[i].x]!='#') {
            ents[i].y--;
        } else
        if (down&&entsl[ents[i].y+1][ents[i].x]==NULL&&map[ents[i].y+1][ents[i].x]!='#') {
            ents[i].y++;
        } else if (right&&entsl[ents[i].y][ents[i].x+1]==NULL&&map[ents[i].y][ents[i].x+1]!='#') {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
            ents[i].x++;
        } else
        if (left&&entsl[ents[i].y][ents[i].x-1]==NULL&&map[ents[i].y][ents[i].x-1]!='#') {
            ents[i].x--;
        } else
        if (up&&entsl[ents[i].y-1][ents[i].x-1]==NULL&&map[ents[i].y-1][ents[i].x-1]!='#') {
            ents[i].y--;
            ents[i].x--;
        } else
        if (up&&entsl[ents[i].y-1][ents[i].x+1]==NULL&&map[ents[i].y-1][ents[i].x+1]!='#') {
            ents[i].y--;
            ents[i].x++;
        } else
        if (down&&entsl[ents[i].y+1][ents[i].x-1]==NULL&&map[ents[i].y+1][ents[i].x-1]!='#') {
            ents[i].y++;
            ents[i].x--;
        } else
        if (down&&entsl[ents[i].y+1][ents[i].x+1]==NULL&&map[ents[i].y+1][ents[i].x+1]!='#') {
            ents[i].y++;
            ents[i].x++;
        } else
        if (left&&entsl[ents[i].y-1][ents[i].x-1]==NULL&&map[ents[i].y-1][ents[i].x-1]!='#') {
            ents[i].y--;
            ents[i].x--;
        } else
        if (left&&entsl[ents[i].y+1][ents[i].x-1]==NULL&&map[ents[i].y+1][ents[i].x-1]!='#') {
            ents[i].y++;
            ents[i].x--;
        } else
        if (right&&entsl[ents[i].y-1][ents[i].x+1]==NULL&&map[ents[i].y-1][ents[i].x+1]!='#') {
            ents[i].y--;
            ents[i].x++;
        } else
        if (right&&entsl[ents[i].y+1][ents[i].x+1]==NULL&&map[ents[i].y+1][ents[i].x+1]!='#') {
            ents[i].y++;
            ents[i].x++;
        }


    entsl[ents[i].y][ents[i].x]=&ents[i];

}

void monattack(ent* mon) {
    putmsg("The Random soul attacks you");
    ents[0].hp-=mon->str;
}


void drawmap() {
    clear();
    for (int y=0;y<YSIZE;y++) {
        for (int x=0;x<XSIZE;x++) {
            if (range(y,x,ents[0].y,ents[0].x)<=ents[0].vis) {
                mvaddch(y,x,map[y][x]);
                seen[y][x]=true;
            } else if (seen[y][x]==true) {
                mvaddch(y,x,map[y][x]|COLOR_PAIR(COLOR_MAGENTA));
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
            }
        }
    }
    for (int i=0;i<numitems;i++) {
        if (range(item[i].y,item[i].x,ents[0].y,ents[0].x)<=ents[0].vis) {
            mvaddch(item[i].y,item[i].x,item[i].sym|COLOR_PAIR(COLOR_YELLOW));
        }
    }
    for (int i=1;i<NMON;i++) {
        if (range(ents[i].y,ents[i].x,ents[0].y,ents[0].x)<=ents[0].vis&&ents[i].alive)
            mvaddch(ents[i].y,ents[i].x,ents[i].sym|COLOR_PAIR(COLOR_CYAN));
    }

    mvaddch(ents[0].y,ents[0].x,ents[0].sym);
    mvprintw(YSIZE,61,"HP <---------->");
    for (int i=0;i<10*ents[0].hp/ents[0].mhp;i++) {
        mvaddch(YSIZE,65+i,'*'|COLOR_PAIR(COLOR_RED));
    }
    mvprintw(YSIZE+1,61,"LV <---------->");
    for (int i=0;i<dlvl;i++) {
        mvaddch(YSIZE+1,65+i,'*'|COLOR_PAIR(COLOR_GREEN));
    }
    mvprintw(YSIZE+2,61,"Exp Level - %d",ents[0].lvl);
    putmsg(" ");
}


bool playerturn() {
    if (ents[0].spd<1) {
        ents[0].spd=1;
    }
    if (luck<-1000) {
        luck=-1000;
    }
    if (luck>1000) {
        luck=1000;
    }
    ents[0].counter+=ents[0].spd;
    if (ents[0].counter<=100) {
        return true;
    }


    turncount++;
    if (ents[0].vis<1&&turncount>500) {
        putmsg("You got pwned by a grue.");
        ents[0].hp=0;
    }
    boredom--;
    if (boredom<=0) {
        bored();
    }
    entsl[ents[0].y][ents[0].x]=NULL;
    bool pinput=playerinput();
    if (pinput) {
        ents[0].counter-=100;
    }
    entsl[ents[0].y][ents[0].x]=&ents[0];
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
    if (ents[0].hp>ents[0].mhp) {
        ents[0].hp=ents[0].mhp;
    }
    return pinput;


}

bool playerinput() {
    input=getch();
    switch (input) {
    case '1':
    case 'b':
        if (entsl[ents[0].y+1][ents[0].x-1] != NULL) {
            pattack(entsl[ents[0].y+1][ents[0].x-1]);
        } else
            if (map[ents[0].y+1][ents[0].x-1]!='#') {
                ents[0].y++;
                ents[0].x--;
            }
        return true;
    case '2':
    case 'j':
        if (entsl[ents[0].y+1][ents[0].x] != NULL) {
            pattack(entsl[ents[0].y+1][ents[0].x]);
        } else
            if (map[ents[0].y+1][ents[0].x]!='#') {
                ents[0].y++;
            }
        return true;
    case '3':
    case 'n':
        if (entsl[ents[0].y+1][ents[0].x+1] != NULL) {
            pattack(entsl[ents[0].y+1][ents[0].x+1]);
        } else
            if (map[ents[0].y+1][ents[0].x+1]!='#') {
                ents[0].y++;
                ents[0].x++;
            }
        return true;
    case '4':
    case 'h':
        if (entsl[ents[0].y][ents[0].x-1] != NULL) {
            pattack(entsl[ents[0].y][ents[0].x-1]);
        } else
            if (map[ents[0].y][ents[0].x-1]!='#') {
                ents[0].x--;
            }
        return true;
    case '5':
    case '.':
        return true;
    case '6':
    case 'l':
        if (entsl[ents[0].y][ents[0].x+1] != NULL) {
            pattack(entsl[ents[0].y][ents[0].x+1]);
        } else
            if (map[ents[0].y][ents[0].x+1]!='#') {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
            ents[0].x++;
         }
      return true;
   case '7':
   case 'y':
      if (entsl[ents[0].y-1][ents[0].x-1] != NULL) {
         pattack(entsl[ents[0].y-1][ents[0].x-1]);
      } else
         if (map[ents[0].y-1][ents[0].x-1]!='#') {
            ents[0].y--;
            ents[0].x--;
         }
      return true;
   case '8':
   case 'k':
      if (entsl[ents[0].y-1][ents[0].x] != NULL) {
         pattack(entsl[ents[0].y-1][ents[0].x]);
      } else
         if (map[ents[0].y-1][ents[0].x]!='#') {
            ents[0].y--;
         }
      return true;
   case '9':
   case 'u':
      if (entsl[ents[0].y-1][ents[0].x+1] != NULL) {
         pattack(entsl[ents[0].y-1][ents[0].x+1]);
      } else
         if (map[ents[0].y-1][ents[0].x+1]!='#') {
            ents[0].y--;
            ents[0].x++;
         }
      return true;
   case ';':
   case '*':
      look();
      return false;
   case'+':
   case ' ':
      action();
      return true;
   case '?':
      help();
      return false;
   case 'Q':
      exit(endwin());
   default:
      putmsg("unknown key");
      return false;
   }
}

void pattack(ent* attackee) {
   if (attackee->alive) {
      putmsg("You attack the random soul");
      attackee->hp -= ents[0].str;
      if (attackee->hp<=0) {
         if (whim[PACIFIST]) {
            luck-=10;
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
            }
            if (whim[SLAUGHTER]) {
               luck+=10;
               ents[0].hp+=5;
            }
            killcount+=1;
            exper+=attackee->lvl;
            if (exper>=ents[0].lvl*10) {
               exper=0;
               levelplayer();
            }
            entsl[attackee->y][attackee->x]=NULL;
            attackee->alive=false;
            attackee->sym='#';
            attackee->x=0;
            attackee->y=0;
            attackee->lvl=0;
         }
      }
   }

void init () {

   srand (time(NULL));
   keypad(initscr(),1);//initializes the keypad and starts curses mode
   start_color();   // init color
   for (int c=0; c<=8; c++) { // init color pairs
      init_pair(c,c,0);
   }


   raw (); /* Lets you read chars as they are typed (no need to wait for <ENTER>)
         * Disables interrupts such as ^C, ^S
         * [cbreak() is like raw(), but ^C stops the program] */

   noecho(); /* Don't echo (show) characters as they are typed */

   curs_set(0); /* 0=don't show cursor */

   nonl(); /* [no newline] Without this, ENTER key generates ^M\n
         * With nonl(), ENTER is just ^M (13)
         * The curses value KEY_ENTER is for "cooked" (not raw) mode. */
}


void putmsg(char newmess[]) {
   static char msg1[60];
   static char msg2[60];
   static char msg3[60];
   if (newmess[0]!=' ') {
      strncpy(msg3,msg2,60);
      strncpy(msg2,msg1,60);
      strncpy(msg1,newmess,60);
   }
   mvaddstr(YSIZE,0,&msg3[0]);
   mvaddstr(YSIZE+1,0,&msg2[0]);
   mvaddstr(YSIZE+2,0,&msg1[0]);
}
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
void genmap(int maptype) {
   switch (maptype) {

   case 0:
      genbasic();
      break;
   case 1:
      gendungeon();
      break;
   case 2:
      gentunnel();
      break;
   case 3:
      genempty();
      break;
   case 4:
      genforest();
      break;
   default:
      putmsg("Call to unknown map");
      genmap(0);
      break;
   }
}

void genbasic() {
   for (int y=0;y<YSIZE;y++) {
      for (int x=0;x<XSIZE;x++) {
         map[y][x]=(y==0 || x==0 || y ==YSIZE-1 || x==XSIZE-1 ||rand()%9==0)?'#':'.';
      }
   }
   upy=rand()%(YSIZE-2)+1;
   upx=rand()%(XSIZE-2)+1;
   downy=rand()%(YSIZE-2)+1;
   downx=rand()%(XSIZE-2)+1;
   map[upy][upx]='<';
   map[downy][downx]='>';

}

void gendungeon() {
   int point[NPOINT][2];
   int p2;

   for (int y=0;y<YSIZE;y++) {
      for (int x=0;x<XSIZE;x++) {
         map[y][x]='#';
      }
   }
   for (int i=0;i<NPOINT;i++) {
      point[i][X]= (rand()%(XSIZE-2))+1;
      point[i][Y]= (rand()%(YSIZE-2))+1;
      map[point[i][Y]][point[i][X]]='.';
   }
   for (int connect = 0;connect<NCONNECT;connect++) {
      for (int i=0;i<NPOINT;i++) {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
        p2=rand()%NPOINT;
        int pointy=point[i][Y],pointx=point[i][X];
        bool up,down,left,right,ul,ur,dl,dr;



        while (pointx != point[p2][X] || pointy != point[p2][Y]) {


          up= (pointy - point[p2][Y]>0);
          down=(pointy - point[p2][Y]<0);
          left= (pointx - point[p2][X]>0);
          right= (pointx - point[p2][X]<0);
          ul=(up&&left);
          ur=(up &&right);
          dl=(down&&left);
          dr=(down&&right);

          if (ur) {
            pointy--;
            pointx++;
          } else
            if (ul) {
              pointy--;
              pointx--;
            } else
              if (dr) {
                pointy++;
                pointx++;
              } else
                if (dl) {
                  pointy++;
                  pointx--;
                } else
                  if (up) {
                    pointy--;
                  } else
                    if (right) {
                      pointx++;
                    } else
                      if (down) {
                        pointy++;
                      } else
                        if (left) {
                          pointx--;
                        }
          map[pointy][pointx] = '.';

        }
      }
    }
    map[point[0][Y]][point[0][X]]='<';
    map[point[NPOINT-1][Y]][point[NPOINT-1][X]]='>';
    upy=point[0][Y];
    upx=point[0][X];
    downy=point[NPOINT-1][Y];
    downx=point[NPOINT-1][X];
}
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
void gentunnel() {
   int point[TUNPOINT][2];
   int p2;

   for (int y=0;y<YSIZE;y++) {
      for (int x=0;x<XSIZE;x++) {
         map[y][x]='#';
      }
   }
   for (int i=0;i<TUNPOINT;i++) {
      point[i][X]= (rand()%(XSIZE-2))+1;
      point[i][Y]= (rand()%(YSIZE-2))+1;
      map[point[i][Y]][point[i][X]]='.';
   }

   for (int i=0;i<TUNPOINT;i++) {
      p2=i+1;
      int pointy=point[i][Y],pointx=point[i][X];
      bool up,down,left,right,ul,ur,dl,dr;


      if (p2<=(TUNPOINT-1)) {
         while (pointx != point[p2][X] || pointy != point[p2][Y]) {


            up= (pointy - point[p2][Y]>0);
            down=(pointy - point[p2][Y]<0);
            left= (pointx - point[p2][X]>0);
            right= (pointx - point[p2][X]<0);
            ul=(up&&left);
            ur=(up &&right);
            dl=(down&&left);
            dr=(down&&right);

            if (ur) {
               pointy--;
               pointx++;
            } else
               if (ul) {
                  pointy--;
                  pointx--;
               } else
                  if (dr) {
                     pointy++;
                     pointx++;
                  } else
                     if (dl) {
                        pointy++;
                        pointx--;
                     } else
                        if (up) {
                           pointy--;
                        } else
                           if (right) {
                              pointx++;
                           } else
                              if (down) {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
                                pointy++;
                            } else
                                if (left) {
                                    pointx--;
                                }
                        map[pointy][pointx] = '.';

                    }
                }
            }
            map[point[0][Y]][point[0][X]]='<';
            map[point[TUNPOINT-1][Y]][point[TUNPOINT-1][X]]='>';
            upy=point[0][Y];
            upx=point[0][X];
            downy=point[TUNPOINT-1][Y];
            downx=point[TUNPOINT-1][X];
        }

        void genforest() {
            int point[TUNPOINT][2];
            int p2;

            for (int y=0;y<YSIZE;y++) {
                for (int x=0;x<XSIZE;x++) {
                    map[y][x]=(y==0 || x==0 || y ==YSIZE-1 || x==XSIZE-1 ||rand()%2==0)?'#':'.';
                }
            }
            for (int i=0;i<TUNPOINT;i++) {
                point[i][X]= (rand()%(XSIZE-2))+1;
                point[i][Y]= (rand()%(YSIZE-2))+1;
                map[point[i][Y]][point[i][X]]='.';
            }

            for (int i=0;i<TUNPOINT;i++) {
                p2=i+1;
                int pointy=point[i][Y],pointx=point[i][X];
                bool up,down,left,right,ul,ur,dl,dr;


                if (p2<=(TUNPOINT-1)) {
                    while (pointx != point[p2][X] || pointy != point[p2][Y]) {


                        up= (pointy - point[p2][Y]>0);
                        down=(pointy - point[p2][Y]<0);
                        left= (pointx - point[p2][X]>0);
                        right= (pointx - point[p2][X]<0);
                        ul=(up&&left);
                        ur=(up &&right);
                        dl=(down&&left);
                        dr=(down&&right);

                        if (ur) {
                            pointy--;
                            pointx++;
                        } else
                            if (ul) {
                                pointy--;
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
                pointx--;
              } else
                if (dr) {
                  pointy++;
                  pointx++;
                } else
                  if (dl) {
                    pointy++;
                    pointx--;
                  } else
                    if (up) {
                      pointy--;
                    } else
                      if (right) {
                        pointx++;
                      } else
                        if (down) {
                          pointy++;
                        } else
                          if (left) {
                            pointx--;
                          }
          map[pointy][pointx] = '.';

        }
      }
    }
    map[point[0][Y]][point[0][X]]='<';
    map[point[TUNPOINT-1][Y]][point[TUNPOINT-1][X]]='>';
    upy=point[0][Y];
    upx=point[0][X];
    downy=point[TUNPOINT-1][Y];
    downx=point[TUNPOINT-1][X];
}

void genempty() {
    for (int y=0;y<YSIZE;y++) {
      for (int x=0;x<XSIZE;x++) {
        map[y][x]=(y==0 || x==0 || y ==YSIZE-1 || x==XSIZE-1)?'#':'.';
      }
    }
    upy=rand()%(YSIZE-2)+1;
    upx=rand()%(XSIZE-2)+1;
    downy=rand()%(YSIZE-2)+1;
    downx=rand()%(XSIZE-2)+1;
    map[upy][upx]='<';
    map[downy][downx]='>';
}

int dice(int numDice,int sideDice,int bonus) {
    int accum=0;
    for (int i=0;i<numDice;i++) {
      accum += rand()%sideDice+1;
    }
    return accum+bonus;
}

double range(int y1,int x1,int y2, int x2) {
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
   double ydif=(double)y1-(double)y2;
   double xdif=(double)x1-(double)x2;
   return sqrt((xdif*xdif)+(ydif*ydif));
}

void clearlocs() {
   for (int y=0;y<YSIZE;y++) {
      for (int x=0;x<XSIZE;x++) {
         entsl[y][x]=NULL;
         seen[y][x]=false;
         if (iteml[y][x]!=NULL) {
            destitem(iteml[y][x]);
         }
      }
   }
}

void genitem(int i) {
   item[i].sym='!';
   do {
      item[i].y=rand()%(YSIZE-2)+1;
      item[i].x=rand()%(XSIZE-2)+1;
   } while (map[item[i].y][item[i].x]!='.'||iteml[item[i].y][item[i].x]!=NULL);
   iteml[item[i].y][item[i].x]=&item[i];
}

void action() {

   if (iteml[ents[0].y][ents[0].x]!=NULL) {
      itemcount+=1;
      ents[0].hp+=5+(5*dlvl);
      putmsg("you drink the potion");
      destitem(iteml[ents[0].y][ents[0].x]);
   }
   if (map[ents[0].y][ents[0].x]=='>') {
      nextlevel();
   } else
      if (map[ents[0].y][ents[0].x]=='<') {
         prevlevel();
      }
   if (map[ents[0].y][ents[0].x]=='*') {
      stone=true;
      putmsg("You have recovered the Stone of Chaos");
      map[ents[0].y][ents[0].x]='.';
   }
   if (map[ents[0].y][ents[0].x]=='_') {
      if (stone) {
         clear();
         mvprintw(3,5,"CONGRATULATIONS!! YOU WIN!!!!!");
         getch();
         getch();
         exit(endwin());
      } else {
         luck-=300;;
      }
   }

}
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
void destitem(items* it) {
   iteml[it->y][it->x]=NULL;
   it->sym='#';
   it->x=0;
}

void putitems() {
   for (int i=0;i<numitems;i++) {
      genitem(i);
   }
}

void nextlevel() {
   if (!noquestion) {
      luck+=5+question();
   }
   noquestion=false;
   if (stone) {
      luck-=70;
   }
   if (whim[GOBACK] &&stone) {
      luck+=150;
   }
   newwhim();

   dlvl++;
   if (dlvl>10) {
      dlvl=10;
   }
   if (!whim[GETOUT]) {
      clearlocs();
      genmap(rand()%5);
      ents[0].y=upy;
      ents[0].x=upx;
      for (int i=1;i<NMON;i++) {
         genmon(i,dlvl);
      }
      if (dlvl==10) {
         map[downy][downx]='*';
      }
      putitems();
   } else {
      getout();
   }
}

void prevlevel() {
   if (!stone) {
      luck-=70;
   }
   if (whim[GOBACK] &&!stone) {
      luck+=170;
   }
   if (!noquestion) {
      luck+=question();
   }
   noquestion=false;
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
    newwhim();
    dlvl--;
    if (dlvl<1) {
        dlvl=1;
    }
    if (!whim[GETOUT]) {
        clearlocs();
        genmap(rand()%5);
        ents[0].y=downy;
        ents[0].x=downx;
        for (int i=1;i<NMON;i++) {
            genmon(i,dlvl);
        }
        if (dlvl==1) {
            map[upy][upx]='_';
        }
        putitems();
    } else {
        getout();
    }
}

int question() {
    clear();
    int rnum=rand()%3+1;
    mvprintw(0,20,"What is the RN God thinking about?");
    for (int i=0;i<3;i++) {
        mvprintw(i+3,0,"%d)",i+1);
        mvaddstr(i+3,4,thought[(rand()%NTHOUGHT)]);
    }
    input=getch();

    switch (input) {
    case '1':
        return(rnum==1)?50:-50;
    case '2':
        return(rnum==2)?50:-50;
    case '3':
        return(rnum==3)?50:-50;
    default:
        putmsg("ANSWER THE QUESTION NEXT TIME!!");
        ents[0].hp=1;
        return-500;
    }
    return 0;
}

void newwhim() {

    switch (wi) {
    case KILLTWO:
        checkkills(2);
        break;
    case KILLSEVEN:
        checkkills(7);
        break;
    case GOFAST:
        checkspeed((turncount<=100));
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
        break;
    case GOSLOW:
        checkspeed((turncount>200));
        break;
    case PACIFIST:
        if (killcount==0)levelplayer();
        break;
    case NOITEMS:
        checkitems(0);
        break;
    case ALLITEMS:
        checkitems(0);
        break;
    default:
        break;
    }
    for (int i=0;i<NWHIMS;i++) {
        whim[i]=false;
    }
    killcount=0;
    turncount=0;
    itemcount=0;
    wi=rand()%NWHIMS;
    whim[wi]=true;
    switch (wi) {
    case KILLTWO:
        putmsg("You feel like killing only a couple people right now");
        break;
    case KILLSEVEN:
        putmsg("7 souls will fall.");
        break;
    case GOFAST:
        putmsg("HURRY UP!!!");
        break;
    case GOSLOW:
        putmsg("Slooooowwww doooowwwwnnn");
        break;
    case PACIFIST:
        putmsg("You feel like a hippy");
        break;
    case SLAUGHTER:
        putmsg("You think genocide mite be a fun thing to do");
        break;
    case NOITEMS:
        putmsg("This water tastes strange");
        break;
    case ALLITEMS:
        putmsg("I'm thirsty");
        break;
    case GOBACK:
        putmsg("Someone is partying behind you");
        break;
    case GETOUT:
        break;
    default:
        putmsg("Error in whim generator");
        break;
    }
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
}

void checkkills(int desired) {
   if (killcount==desired) {
      luck+=50;
   } else
      luck-=75;
}

void checkspeed(bool desired) {
   if (desired) {
      luck+=50;
   } else {
      luck-=70;
   }
}

void checkitems(int desired) {
   if (itemcount==desired) {
      luck+=10;
      numitems*=2;
   } else {
      luck-=50;
      numitems/=2;
   }
   if (numitems>MAXITEM) {
      numitems=MAXITEM;
   }
   if (numitems<1) {
      numitems=1;
   }
}

void getout() {
   if (dlvl==1) {
      wi+=15;
      dlvl--;
      noquestion=true;
      nextlevel();
      return;
   }
   clearlocs();
   genmap(0);
   ents[0].y=upy;
   ents[0].x=upx;
   for (int i=2;i<NMON;i++) {
      ents[i].alive=false;
      ents[i].x=0;
      ents[i].sym='#';
   }
   genmon(1,1);
   ents[1].hp=10000;
   ents[1].spd=ents[0].spd;
   ents[1].str=10000;
   ents[1].vis=200;
   ents[1].sym='&';
   map[upy][upx]='.';
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
    putitems();
    putmsg("DAMN");
    putmsg("A COP!");
    putmsg("RUN!!");
}

int arniemod() {

    if (luck==0||luck/100==0||-luck/100==0) {
        return 0;
    } else if (luck>0) {
        return (rand()%(luck/100));
    } else if (luck<0) {
        return -(rand()%((-luck)/100));
    }
    putmsg("error in arniemod function");
    return 0;
}

void bored() {
    int play=rand()%NPLAYS;
    boredom=dice(30,5,0);
    int nports,times;
    switch (play) {
    case 0:

        putmsg("You are pulled in a  nonexistant direction");
        if ((luck>=0&&!stone)||(luck<0&&stone)) {
            for (nports=rand()%2+1;nports>0;nports--) {
                noquestion=true;
                nextlevel();
            }
        }
        if ((luck<0&&!stone)||(luck>=0&&stone)) {
            for (rand()%2+1;nports>0;nports--) {
                noquestion=true;
                prevlevel();
            }
        }
        break;
    case 1:
        putmsg("The RN God touches you.");
        ents[0].hp+=(arniemod()*(rand()%5+1));
        break;
    case 2:
        putmsg("Somebody gets stronger.");
        for (times=rand()%5+1;times>0;times--) {
            int mon=rand()%NMON;
            if (!ents[mon].alive) {
                break;
            }
            if (mon==0) {
                levelplayer();
            } else {
                levelmonster(mon);
            }
        }
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
      break;
   case 3:
      putmsg("Arnie is bored with you.");
      luck-=dice(3,50,0);
      break;
   case 4:
      putmsg("You amuse the Random Number God");
      luck+=dice(3,50,0);
      break;
   case 5:
      levelplayer();
      break;
   case 6:
      putmsg("Barfff... ugggh");
      demoteplayer();
      break;
   case 7:
      putmsg("A hot wind passes through here");
      for (int i=0;i<NMON;i++) {
         ents[i].hp-=dice(dlvl,20,0);
      }
      break;
   case 8:
      putmsg("Beer crosses your mind");
      numitems+=arniemod();
      if (numitems>MAXITEM) {
         numitems=MAXITEM;
      }
      if (numitems<1) {
         numitems=1;
      }

      break;
   case 9:
      putmsg("You feel a very soothing wind");
      for (int i=0;i<NMON;i++) {
         ents[i].hp+=dice(dlvl,20,0);
      }
      break;
   case 10:
      putmsg("Time seems to be distorted");
      ents[0].spd+=arniemod();
      break;
   case 11:
      putmsg("Your eyes feel weird.");
      ents[0].vis+=arniemod();
      break;
   case 12:
      putmsg("The dead arise");
      for (int j=1;j<NMON;j++) {
         if (ents[j].alive==false) {
            genmon(j,dlvl-arniemod());
         }
      }
      break;
   default:
      break;
```

/* Listing continued on next page…*/

/* Listing continued from previous page */

```
    }
}

void help() {
    clear();
    mvprintw(1,0,"Controls:");
    mvprintw(2,0,"Movement Keys:");
    mvprintw(3,0,"789    yku");
    mvprintw(4,0,"456 or h.l");
    mvprintw(5,0,"123    bjn");
    mvprintw(7,0,"space bar : Action(use stairs, pick up or sacrifice item etc");
    mvprintw(8,0,"semicolon (;) : Look command");
    mvprintw(10,0,"Stuff");
    mvprintw(11,0,"@ : You.");
    mvprintw(12,0,"a-Z : Random Souls that want you dead");
    mvprintw(13,0,"! Healing draught");
    mvprintw(14,0," < and > : stairs up/down");
    mvprintw(15,0,"* : The Chaos Stone Sacrifice it on the altar(_) to win the game");
    getch();
}
```