Rogue Reborn

Lab 03 | Group #6

Team Rogue++

Ian Prins
Or Almog
Mikhail Andrenkov

Purpose and Scope

Purpose

> Develop an enhanced version of the original Rogue (1980)
>
> > Client: UI, Gameplay, and Performance
> >
> > Development: Design, Documentation, and Testing

Scope

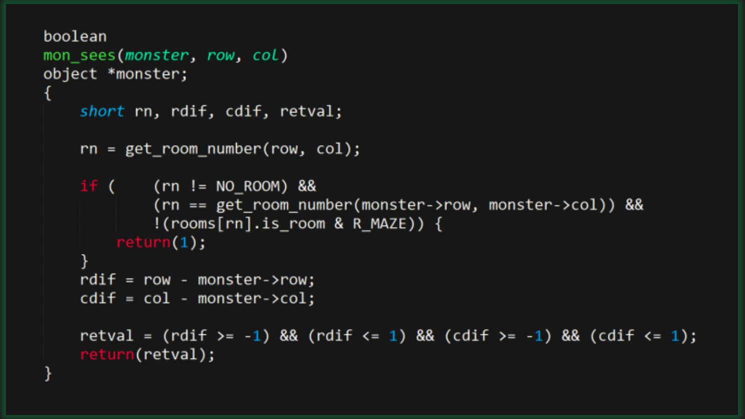
> Implement virtually all functionality present in the original version
>
> Faithful improvements over non-functional qualities such as learnability and aesthetics

Motivation

Development

Strategy

- > Communication: Weekly group meetings
- > Requirements: Avoid unnecessary constraints
- > Authenticity: Reference the original source code when possible

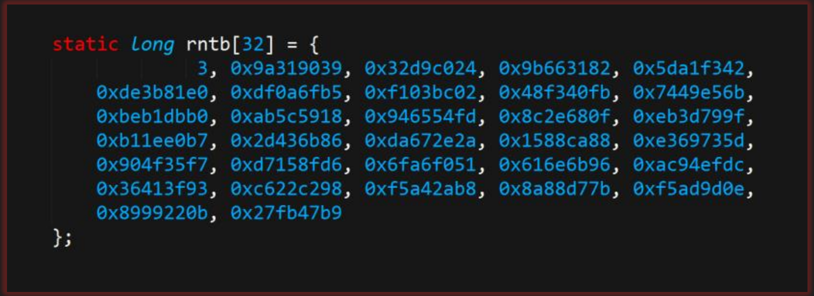


```
boolean
mon_sees(monster, row, col)
object *monster;
{
    short rn, rdif, cdif, retval;

    rn = get_room_number(row, col);

    if (    (rn != NO_ROOM) &&
            (rn == get_room_number(monster->row, monster->col)) &&
            !(rooms[rn].is_room & R_MAZE)) {
        return(1);
    }
    rdif = row - monster->row;
    cdif = col - monster->col;

    retval = (rdif >= -1) && (rdif <= 1) && (cdif >= -1) && (cdif <= 1);
    return(retval);
}
```



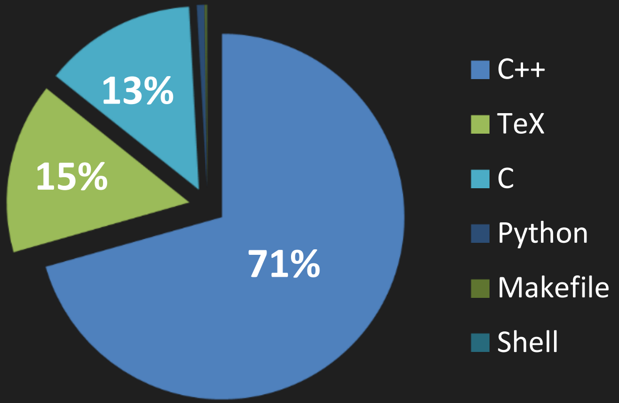
```
static long rntb[32] = {
              3, 0x9a319039, 0x32d9c024, 0x9b663182, 0x5da1f342,
    0xde3b81e0, 0xdf0a6fb5, 0xf103bc02, 0x48f340fb, 0x7449e56b,
    0xbeb1dbb0, 0xab5c5918, 0x946554fd, 0x8c2e680f, 0xeb3d799f,
    0xb11ee0b7, 0x2d436b86, 0xda672e2a, 0x1588ca88, 0xe369735d,
    0x904f35f7, 0xd7158fd6, 0x6fa6f051, 0x616e6b96, 0xac94efdc,
    0x36413f93, 0xc622c298, 0xf5a42ab8, 0x8a88d77b, 0xf5ad9d0e,
    0x8999220b, 0x27fb47b9
};
```

Coding Technologies

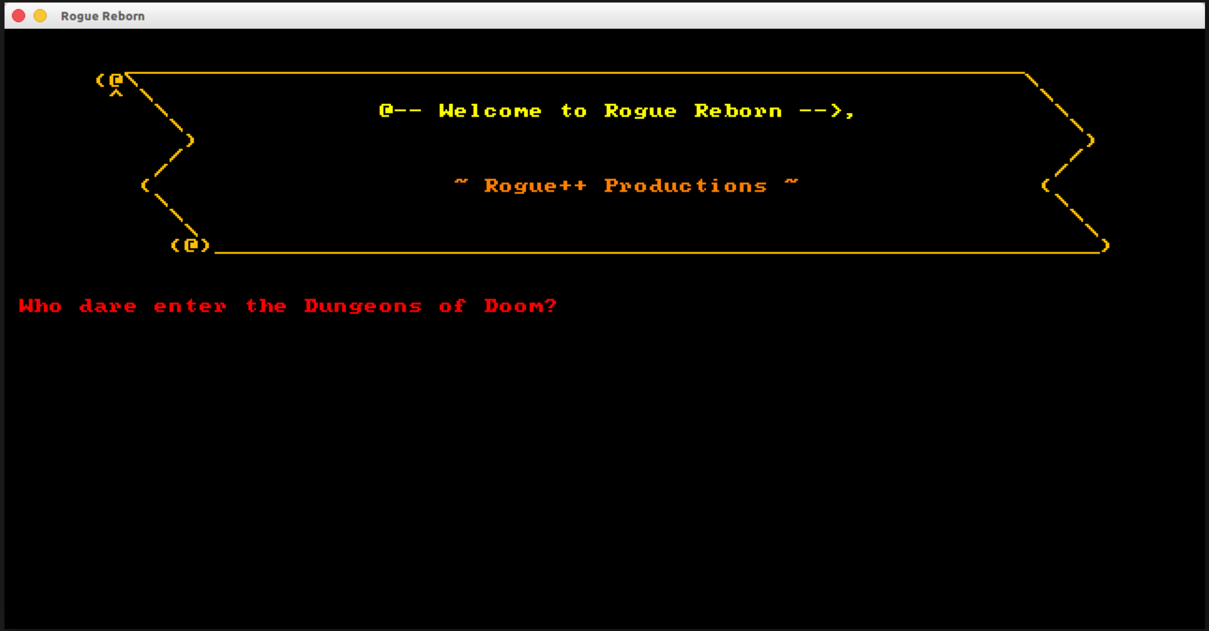
C++ > Efficient, object-oriented, and industry standard (8000+ lines)

Libtcod > Graphics library for emulating a roguelike experience

Other > CI (Continuous Integration)
Git
GDB (GNU Debugger)
Make
Valgrind (Memory Profiler)
g++ dependency detection



Demonstration



```
@-- Welcome to Rogue Reborn -->,

~ Rogue++ Productions ~

Who dare enter the Dungeons of Doom?
```

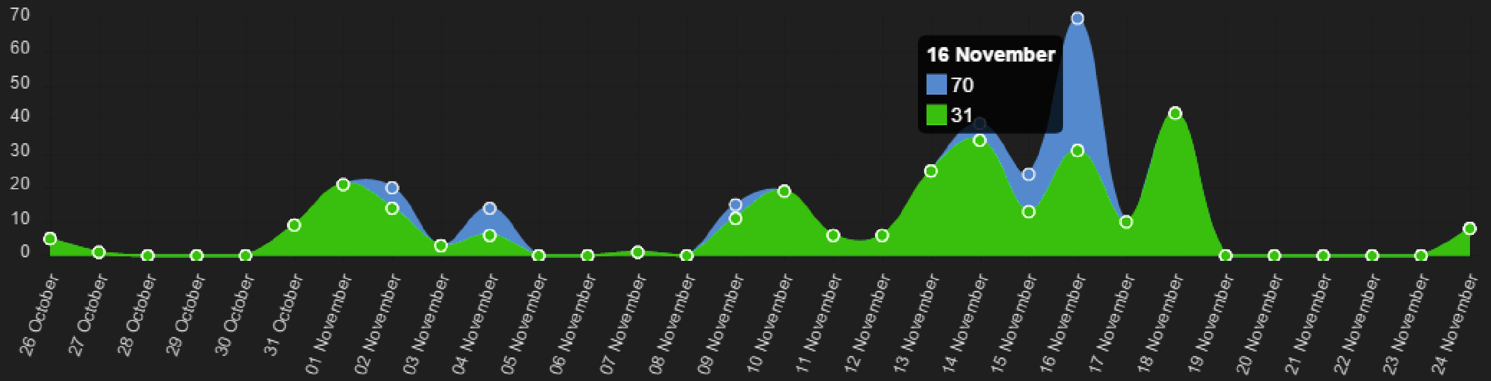
The R^2 Touch

Robustness

> Enabled CI to ensure dysfunctional code changes are flagged

> Ensured compilation did not generate warnings

> Used profiler tools to analyze memory leak performance



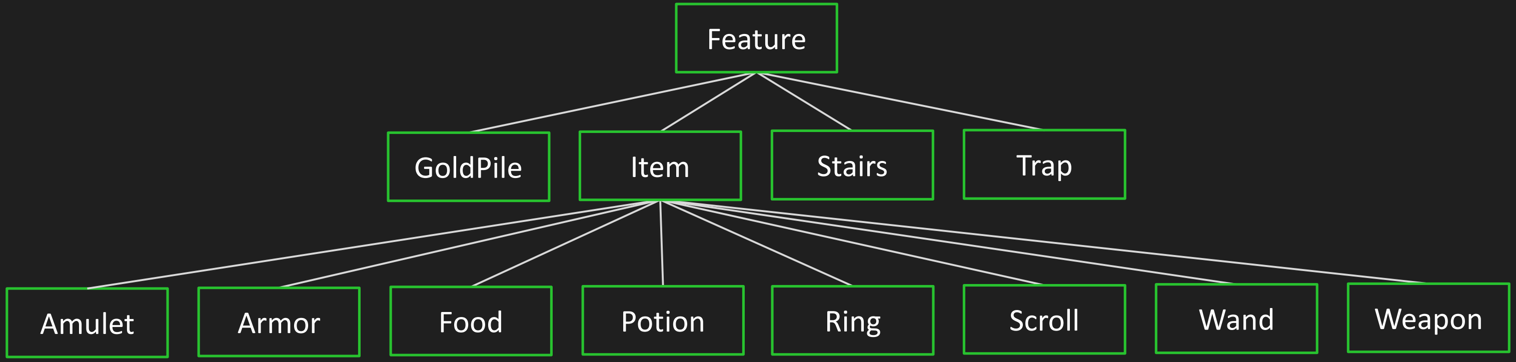
The R^2 Touch Cont'd

Maintainability

> Designed header files to hide implementation secrets

> Generated comprehensive documentation for all modules

> Used GitLab issue tracking to document bugs and known issues



Final Remarks

Project

- Rogue Reborn was successful in establishing an authentic and entertaining Rogue experience
- Success was enabled by frequent contributions (800+ commits), regular communication, and honest feedback
- Application of the Rational Design Process and development tools facilitated the development of maintainable and robust software

Next Steps

- Implement audio features; prepare game for distribution

Questions