David Bacher-Hicks
CS 162
5 December 2016

Final Project

I. Introduction

The final project for CS 162 requires the development of a text-based game. This implementation of a text game can be considered a roguelike game and is largely inspired by earlier roguelike games and the core Dungeons and Dragons game mechanics (v. 3.0). The game will render floors of a dungeon for the player to explore and that dungeon is occupied by monsters (*mobs*) that the player will have to fight or avoid. There will be items to collect, including keys to unlock doors and equipment that may aid the player in defeating monsters. Some of these may just be found lying around, while others may be dropped by monsters after they have been killed.

Defeating monsters will award the player experience and a chance to collect dropped items. After having gained a certain amount of experience, the player will *level up*, which in this game will mean an increase in total health points and base attack modifier, an additive modifier to the player's chance to hit an enemy when attacking.

The concept of the game is that the Player is an adventurer in a fantasy/RPG setting who has taken on the task of eliminating a minotaur that has taken up residence in a dungeon. This task has a time limit of 5 days and days will advance in one-day increments when the player rests to recover health. Resting is the only way to recover health, which will be depleted by monster attacks.

The player has been given generous stats for a level one character, but Dungeons and Dragons is an unforgiving game and is meant to be played with more than one character. As a result, this game is a little bit tougher than I had intended. A few unlucky die rolls can spell disaster, especially if the player hasn't had the chance to level up very much to increase their pool of hit points.

To alleviate this difficulty and allow for easier testing, a hidden room has been added with extremely good equipment. See the game guide at the end of this document for instructions on how to find and obtain this equipment, as well as general gameplay information.

II. Global Constants and Enumerated Data

1. Global Constants

The game uses a number of global constants. See Appendix A for a listing of the global constants.

2. Enumerated Data Types

Table 1. Enumerated data types.

Name	Values	Location Defined
ability	{ STR, DEX, CON, INT, WIS, CHA }	Character.hpp
direction	{ UP, RIGHT, DOWN, LEFT }	Space.hpp

3. Structs

3.1. mob_data

mob_data is a structure that contains the information necessary to instantiating a particular type of monster. mob_data structures are created when the monster data file is loaded at game start up and are written to a map, Game::mobs, keyed by the mob_ID string.

Table 2. mob_data data elements.

Name	Data Type	Use
id	std::string	A unique string identifying the monster type. Key to the mob.
name	std::string	A display name; not necessarily unique.
render_char	char	The character that the game will display to represent the mob.
ac	int	The mob's armor class.
hp	int	The mob's hit points.
die_n	int	The number of damage dice.
die_s	int	The number of sides for each damage die.
die_m	int	An additive modifier to damage die rolls.
b_atk	int	The mob's base attack bonus.
cr	double	The challenge rating of the mob; used to determine experience points awarded for defeating.

3.2. attack_data

attack_data is a structure that contains the information associated with an attack event that occurs when one Character attacks another. It contains a damage and attack component, with attack meaning likelihood to hit.

Table 3. attack_data data elements.

Name	Data Type	Use
attack_roll	int	The likelihood that the attack will hit its target.
damage_roll	int	The damage inflicted if the attack hits.

III. Design

1. Character Class and its Subclasses

Table 4. Character class member data.

Name	Data Type	Use
name	std::string	The name of the character.
render_char	char	The character to render.
coord	Coord	The location of the character.
inventory	std::map <Item *, int>	A map of the items the character is carrying, with item pointers mapped to the quantity carried.
hp	int	The character’s current hp.
max_hp	int	The character’s maximum hp.
b_atk	int	The character’s base attack bonus.

1.1. Character

1.1.1. Methods

a. Character

Constructor of the Character class.

b. ~Character

Destructor of the Character class.

c. get_render_char : *char*

Returns the character’s render character.

d. set_coord(int x, int y)

   Sets the character's location to a Coord(x, y).

e. set_coord(Coord coord)

   Sets the character's location to the passed coord.

f. get_coord() : Coord

   Returns the character's location.

g. has(Item *item) : bool

   Returns whether the character has the passed item.

h. item_count(Item *item) : int

   Returns the quantity held of the passed item.

i. add_item(Item *item)

   Adds the passed item to the character's inventory.

j. remove_item(Item *item) : bool

   Removes the passed item from the inventory. This is either a decrement of the mapped quantity by 1 or deletion of the entry if the quantity is reduced to 0. Returns true if the item was successfully removed.

k. set_name(std::string name)

   Sets the character's name to the passed value.

l. get_name() : std::string

   Returns the character's name.

m. get_hp() : int

   Returns the character's current hp.

n. rest()

   If the floor has been cleared, advances game time by one day and restores the character's health to the maximum hp value.

o. get_max_hp() : int

Returns the character’s maximum hp.

p. is_dead() : bool

Returns whether the character has died (current hp <= 0);

q. *Pure virtual* attack() : attack_data
r. Pure virtual defend() : bool

s. get_inventory() : map<Item *, int> *

Returns a pointer to the character’s inventory.

1.2. Player

Table 5. Player class member data.

Name	Data Type	Use
ability_score	std::map <ability, int>	Map of ability scores to their values.
equipped_weapon	Item*	The item the character has equipped.
equipped_armor	Item*	The armor the character has equipped.
experience	experience	The total experience the character has earned.
level	int	The player’s current level.

1.2.1. Methods

a. Player(std::string = “”)

Constructor for the Player class. Initializes all ability scores, initial HP, level base attack bonus and experience points.

b. get_ability_mod(ability ab) : int

Returns the modifier associated with the passed ability score.

c. get_ability(ability ab) : int

Returns the passed ability score.

d. inc_ability(ability ab) : void

Increments the passed ability score.

e. carry_weight() : double

   Returns the total weight of items carried.

f. max_carry() : double

   Returns the maximum weight the character can carry.

g. encumbered() : bool

   Returns whether the character's weight exceeds the maximum. At this point, the character can no longer move.

h. equip_item(Item *item)

   Equips passed weapons and armor.

i. get_weapon() : Item *

   Returns the equipped weapon.

j. get_armor() : Item *

   Returns the equipped armor.

k. add_experience()

   Adds experience and increments a character's level if appropriate.

l. get_experience() : int

   Returns the character's total accumulated experience.

m. get_level() : int

   Returns the character's level.

n. get_b_atk() : int

   Returns the character's base attack.

o. attack() : attack_data

   Creates and returns an attack_data structure based on the character's equipped weapon, appropriate status modifiers and base attack bonus.

p. defend(attack_data) : bool

Determines if incoming attacks hit or miss, adjusting current HP as necessary.

1.3. Mob

Table 6. Mob class member data.

Name	Data Type	Use
damage_die	Die*	Die rolled to determine damage component of attack
ac	int	Innate AC.
cr	int	Challenge rating; relates to difficulty and is used to calculate experience value.

1.3.1. Methods

a. Mob(mob_data *data, Coord coord)

Constructor for the Mob class creates a Mob with the passed mob_data structure and the passed coord.

b. ~Mob

Destructor for the Mob class deallocates the damage_die.

c. attack() : attack_data

Creates and returns an attack_data structure with the base attack bonus and damage die member object.

d. defend(attack_data atk) : bool

Determines if incoming attacks hit or miss and adjusts HP as appropriate.

e. get_experience() : int

Returns the amount of experience the mob is worth.

2. Space Class and its Subclasses

Table 7. Space class member data.

Name	Data Type	Use
items	std::vector <Item*>	Items present in the space.
present_character	Character*	The character present in the space.
coord	Coord	The coordinates of the space.
render_char	char	The character to render for the space.
linked_spaces	std::map <direction, Space*>	A map of linked spaces, using the enum, direction, as the key.

2.1. Space

2.1.1. Methods

a. Space(Coord coord)

Initializes the space with the passed coord and sets all linked spaces in { UP, DOWN, LEFT, RIGHT } and the present character to NULL.

b. Space(int x, int y)

Initializes the space with a Coord(x, y) and sets all linked spaces in { UP, DOWN, LEFT, RIGHT } and the present character to NULL.

c. *virtual* ~Space()

Destructor; deallocates any dynamic objects present in the space.

d. *virtual* passable() : bool

Returns whether the space is passable. False if it blocks.

e. *virtual* is_locked() : bool

Returns whether the space is locked.

f. link(Space *space, direction dir)

Links the passed space with this space, using the passed direction as a key to the member map, linked_spaces.

g. get_character() : Character*

   Returns the character present in the space.

h. add_character(Character *character)

   Adds a character to the space.

i. delete_character()

   Removes a character from the space.

j. remove_character() : Character*

   Removes and returns the present character.

k. get_items() : std::vector<Item*>*

   Returns a pointer to the space's items vector.

l. *virtual* add_item(Item *item) : bool

m. Adds an item to the space's items vector.

n. remove_item(std::string item_id) : Item*

   Searches the items vector for the passed id and removes the item if it exists.

o. set_rander_char(char c)

   Sets the render character to the passed character.

p. *virtual* get_render_char() : char

   Returns the render character.

q. get_coord() : Coord

   Returns the coordinates of the space.

r. x() : int

   Returns the x-coordinate of the space.

s. y() : int

   Returns the y-coordinate of the space.

2.2. OpenSpace

The OpenSpace class is a subclass of the Space class and has no member data of its own aside from those data members inherited from the Space class. Floors are most likely to be made up of open space and therefore most spaces on any given floor are most likely to be OpenSpace objects.

2.2.1. OpenSpace Methods

a. OpenSpace(Coord)

Constructor initializing the coord to the passed coord.

b. OpenSpace(x, y)

Constructor initializing the coord to Coord(x,y)

c. virtual passable()

An OpenSpace is passable by default, but will search through the occupying objects for blocking objects and return false if any are found.

d. virtual get_render_char() : char

Returns the render character for open space.

e. virtual add_item(Item *item) : bool

Adds the passed item, returning true if the item was added successfully.

f. virtual remove_item(std::string item_ID) : Item*

Removes the item passed, identified by item ID.

2.3. Wall

The Wall class is a subclass of the Space class and has no member data of its own aside from the data members inherited from the Space class. The purpose of the wall class is to serve as general purpose walls that block character movement and partition space on a floor.

2.3.1. Wall Methods

a. Wall (Coord coord)

Constructor initializing coord to the passed coord.

b. Wall (int x, int y)

Constructor initializing coord to Coord(x, y)

c. passable() : bool

Walls are impassible by default, so passable always returns false.

2.4. Door

Table 8. Door class member data.

Name	Data Type	Use
is_open	bool	Keeps track of door state: true – open, false - closed
keyID	std::string	The item_ID of the key that opens the door, if locked. Holds an empty string if the door is not locked.

2.4.1. Door Methods

a. Door(Coord coord)

Constructor initializing coord to the passed Coord.

b. Door(int x, int y)

Constructor initializing coord to Coord(x,y).

c. *virtual* passable() : bool

In the context of a door, this value is tied to the door's state and returns true if open, false if closed.

d. *virtual* get_render_char() : char

Returns the render character for the door. This also varies depending on the state of the door, returning the character corresponding to an open door if open and the character corresponding to a closed door if closed.

e. *virtual* is_locked() : bool

Returns whether the door is locked.

f. key_str() : std::string

Returns the item_id of the key that will unlock the door.

g. set_key(std::string key_id)

   Sets the item_id of the key that will unlock the door.

h. open() : bool

   Opens the door.

i. close() : bool

   Closes the door.

2.5. SecretDoor

The SecretDoor class is a subclass of Space, providing one additional data member: is_open. The purpose of a SecretDoor is to provide door functionality without being immediately apparent to the player that there is a door. The render character for a secret door is the same as a wall while it is closed, therefore blending into walls unless activated and opened by the player.

2.5.1. SecretDoor Methods

a. SecretDoor(Coord coord)

   Constructor initializing coord to the passed coord.

b. SecretDoor(int x, int y)

   Constructor initializing coord to Coord(x,y).

c. virtual passable() : bool

   As in Door, passability is tied to open state. Passable returns true when the SecretDoor is open, false when closed.

d. virtual get_render_character() : bool

   Returns the render character for the SecretDoor, which is a wall character when closed, but an open space character after open.

e. open() : bool

   Opens the SecretDoor.

2.6. Stair

The Stair class is a subclass of Space, introducing two additional data members. The purpose of the Stair class is to serve as a link between floors. When the player traverses a stair, they are transported to another floor, to the location of the linked stair.

Table 9. Stair class member data.

Name	Data Type	Use
linked_coord	Coord	The coordinate of the linked stair.
linked_floor_ID	std::string	The floor_ID of the floor on which the linked stair occurs.

2.6.1. Methods

a. Stair(Coord coord)

Constructor initializing coord to the passed Coord.

b. Stair(int x, int y)

Constructor initializing coord to Coord(x,y).

c. set_linked_coord (Coord coord)

Sets the linked coord tot he passed Coord.

d. set_linked_floor_ID (std::string floor_id)

Sets the linked floor id to the passed string.

e. get_linked_floor_ID() : std::string

Returns the linked floor ID.

f. get_linked_coord(): Coord

Returns the linked Coord.

g. passable()

Returns true.

2.7. UpStair and DownStair Classes

The UpStair and DownStair classes are subclasses of the Stair class. They exist mainly to differentiate for rendering.

3. Floor Class

The Floor class encapsulates and manages a set of spaces and mobs.

Table 10. Floor class member data.

Name	Data Type	Use
spaces	std::map <Coord, Space*>	Maps coordinates to the Floor's spaces.
mob_list	std::set <Character*>	Tracks the mobs present on the floor. The set is iterated through once during each mob movement cycle.

3.1. Floor Methods

3.1.1. ~Floor()

Iterates through spaces and mob_list, both of which are containers of dynamically allocated memory objects, and frees the associated memory.

3.1.2. get_space(int x, int y) : Space*

Returns the space at the passed coordinates.

3.1.3. get_space(Coord coord) : Space*

Returns the space at the passed Coord.

3.1.4. add_space(Space *, int x, int y)

Adds the passed space to spaces at Coord(x,y).

3.1.5. load_floor(std::string path)

Loads a floor from a map layout file located at the passed path.

3.1.6. render_floor() : std::string

Renders the floor to a string, which is returned.

3.1.7. interpret_space(char space_char, Coord coord)

Interprets a space based on the character as it would appear in a map layout file. Creates a new space of the appropriate type at the passed Coord, which is returned. Used by load_floor in populating a new space for each character read.

3.1.8. add_char(Character *char, Coord coord)

Adds a the passed character to the space at the passed Coord.

3.1.9. move_char(Coord from, Coord to)

Moves the character occurring at ***from*** to the space at ***to***.

3.1.10. list_mob(Character *mob)

Adds the passed mob to the mob_list.

3.1.11. unlist_mob(Character *mob)

Removes the passed mov from the mob_list. Called when a mob has been killed prior to the Floor destructor having been called.

3.1.12. get_mob_list () : std::set<Character*>*

Returns a pointer to the mob_list.

3.1.13. get_spaces() : std::map<Coord, Space*>*

Returns a pointer to spaces.

4. Item Class and its Subclasses

The item class and its subclasses are used to encapsulate information about objects that characters can carry in in their inventories. Generic Item objects include keys, while more specialized items include weapons and armor, each with a subclass of its own.

4.1. Item

Item class member data.

Name	Data Type	Use
is_passable	bool	False if the item would obstruct movement to a degree that would render the space it occupies impassible.
item_ID	std::string	A unique string identifying the item.
item_name	std::string	A short name for use in identifying the item in-game.
item_description	std::string	A descriptive phrase, printed on examining the item.
item_weight	double	The weight of the item in lbs.
item_value	double	The value of the item.

4.1.1. Item Methods

a. Item (…)

Constructor initializing the item data to the passed values.

b. passable() : bool

Returns passable.

c. id() : std::string

Returns the item id.

d. description() : std::string

Returns the item description.

e. name() : std::string

Returns the item name.

f. weight() : double

Returns the item weight.

g. value() : double

   Returns the item value.

4.2. Armor

The Armor class is a subclass of Item and represents an item that the Player may equip to enhance their AC statistic to help in avoiding hits in combat. The Armor class has a single data member in addition to those inherited from Item: an integer value for AC.

4.2.1. Armor methods

a. Armor(…)

   Constructor, initialzing the Armor object with the passed values.

b. get_ac() : int

   Returns the Armor's AC.

4.3. Weapon

The Weapon class is a subclass of Item and represents an item that the Player may equip, The equipped item determines most of the damage associated with a Player's attack; however, the Player's strength ability modifier is added to the roll. The Weapon class has a single data member in addition to those inherited from Item: damage_die*, a pointer to Die object that is rolled to determine a component of the Player's attack damage.

4.3.1. Weapon methods

a. *virtual* ~Weapon()

   Deallocates the dynamic damage_die object.

b. roll_damage() : int

   Rolls and returns the result of the damage_die.

5. Game Class

Game class member data.

Name	Data Type	Use
messages	std::vector <std::string>	Accumulates messages to be printed with the map rendering, such as attack results and the results of movement attempts.
floors	std::map <std::string, Floor*>	Contains the floors loaded, using the floor_ID as a key.
items	std::map <std::string, Item*>	Contains the items loaded, using the item_ID as a key. Each item is only instantiated once, with Characters and Spaces containing only pointers to the items coupled with quantities.
mobs	std::map <std::string, mob_data*>	Contains the mob_data structures loaded, containing the information necessary to instantiate a mob of a particular kind. Mobs are instantiated many times, as needed.
player	Player	The player character.
in_progress	bool	Keeps track of whether the game is still in progress.
current_floor	Floor*	A pointer to the floor where the player currently is.
logfile	std::ofstream	A logfile kept open to log events as needed.
days_passed	int	Keeps track of the days spent in the dungeon to enforce a time limit.

5.1. Game Methods

5.1.1. Game

Constructor for the game, completes various setup activities:

1. Loads items
2. Loads mobs
3. Loads mobs
4. Loads floors
5. Links spaces
6. Initializes the player object
7. Initializes the game state (in progress, days spent = 0)

5.1.2. ~Game

Iterates through floors, items and mobs, deleting the dynamically allocated contents.

5.1.3. load_floors

Loads all map files located in the map root folder, instantiating all items and mobs in the floor according to the map data file. Inserts the created floor into the floors map.

5.1.4. link_spaces

Links each space to its adjacent spaces.

5.1.5. load_items

Loads all items from the item data file, inserts into the items map.

5.1.6. load_mobs

Loads all mob data from the mob data file. Inserts into the mobs map.

5.1.7. move_player(direction dir)

Attempts to move the player in the passed direction, but contextually performs other actions as well. If the specified space contains a mob, the request branches to an attack. If the specified space contains a door, the request branches to a door open attempt.

Additional checks are performed after the move to determine if the player should be notified of items present in the space or if the player needs to be transported via a stair to another floor.

move_player triggers a mob movement cycle.

5.1.8. manage_player_inventory

Displays the players inventory and accepts input for inventory commands, including dropping, equipping and examining items.

5.1.9. get_player_inventory_selection() : Item*

Displays the player’s inventory and gets a selection of inventory item, and returns a pointer to the selected item.

5.1.10. print_player_inventory

Prints the player’s inventory.

5.1.11. player_get_items()

Attempts to get an item from the Player’s current space, removing it from the space and adding it to the Player’s inventory.

5.1.12. player_drop_item

Gets a selection and then attempts to drop the item, removing it from the character’s inventory and adding it to the character’s current space.

5.1.13. player_examine_item

Gets a selection and then prints the description of the item.

5.1.14. player_equip_item

Gets a selection and then attempts to equip the selected item. If an inappropriate item (non armor and non weapon) is chosen, a message is displayed notifying the user.

5.1.15. player_attack_mob(Character *mob)

Performs an attack on the passed mob. Passes a Player.attack() to mob->defend() and displays a message regarding the result.

5.1.16. player_rest

Rests the player, recovering to full health. This will advance the game time by one day and can only be done after a floor has been cleared of monsters.

5.1.17. print_player_character_sheet

Prints information about the player character, including level, stats and experience.

5.1.18. move_mobs

Iterates through the mobs on the Player’s current floor, paths them toward the player, and conducts attacks if they are within range. If dx or dy are non-zero, movement in that direction is considered priority. For instance, if the player is above the monster to the left, both up and left are considered priority moves.

Priority moves are passed to mob_make_moves and if no moves are made, the remaining directions are passed as well, adding a certain degree of wandering if the mob cannot path toward the player.

5.1.19. mob_make_moves(vector<direction>) : bool

Random directions are attempted until all have been tried or a move is made. Returns whether a move was made.

5.1.20. mob_attack_player(Character *mob) : bool

Performs an attack of the passed mob on the Player. Pushes the result to messages.

5.1.21. read_input (char c)

Interprets user input and branches to the appropriate function.

5.1.22. inc_day

Increments the game day and checks if the time has run out.

5.1.23. Render() : std::string

Renders the current floor, appends the status bar and any stored messages, and returns a string containing the rendering.

5.1.24. print_status_bar() : std::string

Creates and returns a text status bar with information about the player.

5.1.25. coord_from_direction(Coord coord, direction dir)

Returns the coordinate offset from the passed coordinate one space in the passed direction. For example coord_from_direction(Coord(0,0), RIGHT) would return Coord(1,0).

5.1.26. is_in_progress() : bool

Returns whether the game is still in progress. Used by the main loop in the client as a loop condition.

IV. Testing Plan

1. Input
    - 1.1. Movement
        - 1.1.1. Empty spaces
            - a. Movement to empty spaces will result in the Player being moved to that space.
        - 1.1.2. Into walls
            - a. Movement to wall spaces will result in a message being displayed notifying the user that the move cannot be made.
        - 1.1.3. Into enemies
            - a. Movement into enemies will result in an attack being made on the enemy.
        - 1.1.4. Into doors
            - a. Movement into doors will result in an attempt to open the door. The results will depend on whether or not the door is locked, and if the door is locked, the attempt will also depend on whether or not the player has the appropriate key.
            - b. Locked
                - 1. Player has the key
                    - 1. The door will be opened
                - 2. Player does not have the key
                    - 1. The door will not be opened
            - c. Not Locked
                - 1. The door will be opened
        - 1.1.5. Onto items
            - a. A message will be displayed, showing that there are items present that can be picked up.
        - 1.1.6. Onto stairs
            - a. The player will be transported to the linked space.
        - 1.1.7. When encumbered
            - a. The player will not be moved and a message will be displayed notifying the user that they cannot move because they are encumbered.
    - 1.2. Opening Inventory System
        - 1.2.1. The inventory system will be opened.
    - 1.3. Opening Character Sheet
        - 1.3.1. The character sheet will be displayed
    - 1.4. Resting
        - 1.4.1. Room is not empty
            - a. A message will notify the user they cannot rest when there are monsters nearby
        - 1.4.2. Room is empty
            - a. The day will be incremented and the player’s health will be replenished. If the day is greater than the maximum number of days (5) then the game will end.

2. Inventory System
    - 2.1. Drop item
        - 2.1.1. Item is equipped
            - a. A message will be displayed, notifying the user that they cannot drop equipped items.
        - 2.1.2. Item is not equipped

a. The item will be dropped and added to the Player's current space, but the effect on the inventory will depend on the quantity held:

b. Quantity held is 1

   1. The item will be removed from the inventory map

c. Quantity held is greater than 1

   1. The quantity held will be decremented

2.2. Equip item

2.2.1. Item is equipment (armor/weapon)

a. The item will be equipped

2.2.2. Item is not equipment

a. A message will notify the user that they cannot equip the item

2.3. Examine item

2.3.1. The description of the item will be displayed

2.4. Return to play

2.4.1. The user will be returned to the main game screen

3. Game Over Conditions

3.1. Running out of time

3.1.1. If the day is incremented to 6, the game will end.

3.2. Losing in combat (dying)

3.2.1. If the player's health is reduced to 0, the game will end.

3.3. Winning (killing the minotaur)

3.3.1. If the player kills the minotaur, the game will end.

4. Memory Management

The game makes heavy use of dynamic memory management and is therefore susceptible to memory leaks. Testing will be conducted using Valgrind in order to detect any potential memory leaks.

V. Reflection

Development occurred in general accordance with the initial design, although a number of unforeseen data members and methods were added during development. In addition to these inclusions, the scope of the initial design was reduced significantly during development. It was originally planned to implement character creation allowing multiple character class types (per the Dungeons and Dragons 3.0 ruleset) and implementing a much larger portion of the core ruleset. It quickly became apparent that there was not sufficient time to implement the game with such a large scope.

Character creation was reduced to a single character type, which is essentially a fighter with very high initial stats, and the full implementation of the Dungeons and Dragons core rule set was reduced to a limited patchwork of gameplay mechanics, including the melee combat attack and damage rolls. Skills and abilities were dropped entirely and the leveling system was pared down to influencing HP and base attack bonus only.

All tests yielded the expected test results. No memory leaks were detected by Valgrind.

VI. Game Guide

1. Floor map rendering

The game is character rendered and may be difficult to interpret without an explanation of the characters used. The following table lists the meaning of the various render characters.

Character	Meaning
.	An empty space
*	A space with an item
#	A wall
[	A closed door
\	An open door
@	Your character
v	A staircase leading down
^	A staircase leading up
s, S, g, o, b, M	Various monsters

2. Controls

From the main game screen, you may issue commands by pressing keys:

Q – quit the game. **Note that this is a capital Q and not lowercase.**
w / up arrow – move up
a / left arrow – move left
s / down arrow – move down
d / right arrow – move right
r – rest. This will allow your character to rest and recover health.
c – open the character sheet, showing information about your character
i – open your inventory. This will open an inventory management system. Follow the on-screen instructions to equip, drop and examine items. You have a weight limit, and will not be able to move after you have exceeded this limit. If this happens, you will have to drop items in order to move again.

3. Attacking Enemies and Opening Doors

In order to attack enemies or attempt to open doors, issue a move command in the direction of your target. The game will interpret the move as an attack or an attempt to open the door. Some doors are locked. You will automatically search for the key and unlock the door if you have the correct key.

4. Leveling up

   Defeating enemies will earn your character experience. After having earned a certain amount of experience, your character will level up, gaining health points and base attack bonus points.

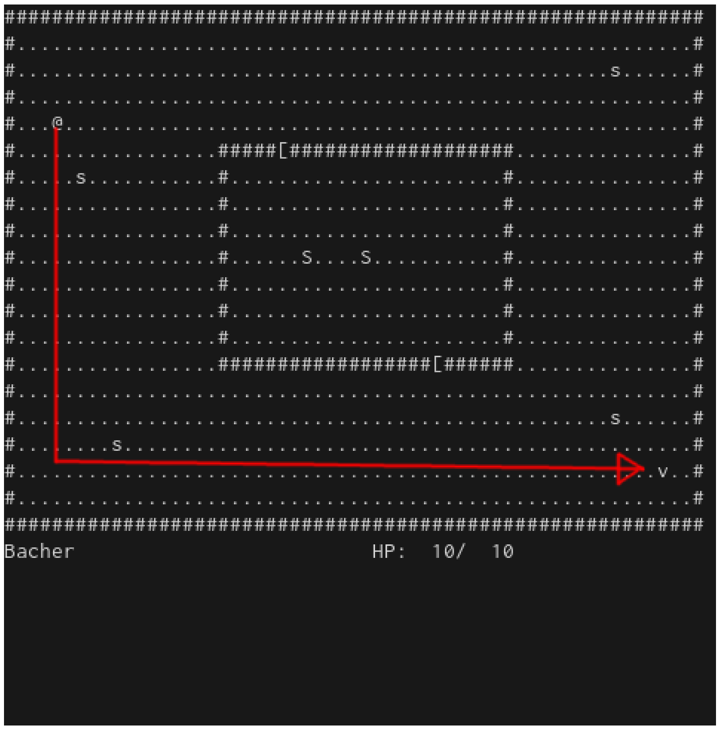
5. Resting

   Your character can rest to recover health points, but this will consume a day of game time and can only be done after a floor has been cleared of enemies. This is the only way to recover lost health points, but remember that you are on a time limit of 5 days, so don’t rest too frequently, or else you may run out of time and lose the game.

6. Inventory Management

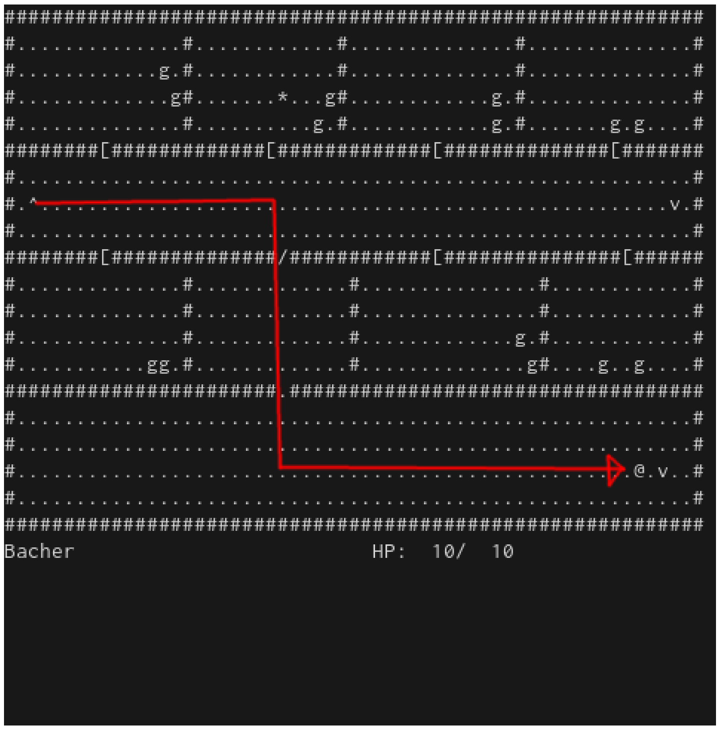
   You can bring up an inventory management system from the main screen by pressing i. From this screen, you can equip, drop and examine items. Your carry limit and total carried weight are also displayed on this screen. If you are carrying too much weight, you will not be able to move and will have to drop items in order to regain the ability to move.

7. Image Walkthrough



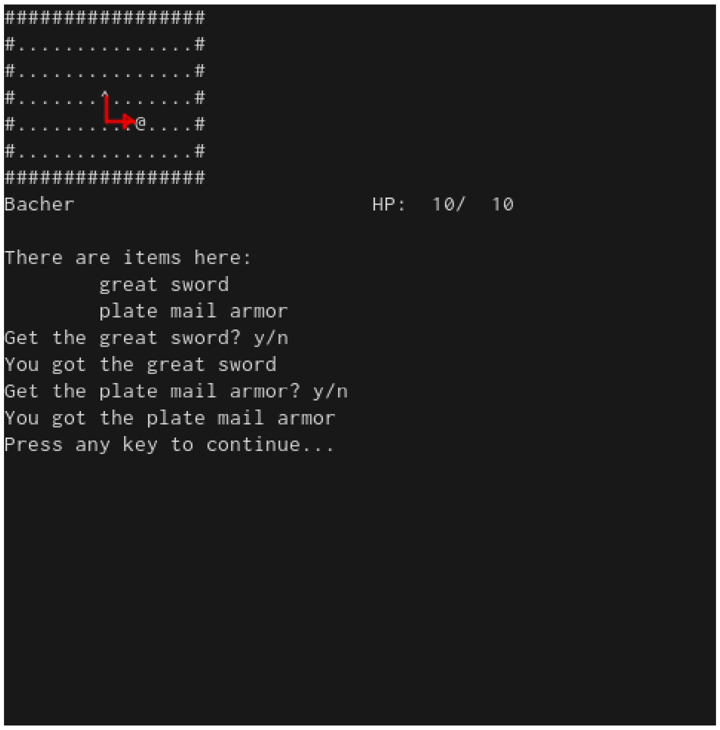
```
###########################################################
#.........................................................#
#..................................................s......#
#.........................................................#
#...@.....................................................#
#.................#####[###################...............#
#.....s...........#.......................#...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................#......S....S...........#...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................##################[######...............#
#.........................................................#
#..................................................s......#
#........s................................................#
#......................................................v..#
#.........................................................#
###########################################################
Bacher                         HP:  10/  10
```

1. First floor: optionally, fight the skeletons, then proceed down the stairs to the second floor.



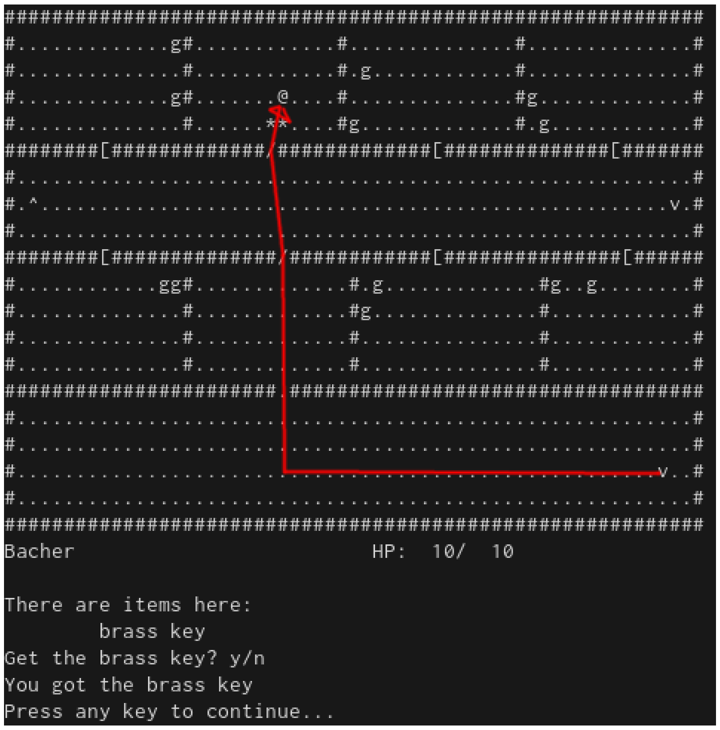
```
###########################################################
#..............#............#..............#..............#
#............g.#............#..............#..............#
#.............g#.......*...g#............g.#..............#
#..............#..........g.#............g.#.......g.g....#
########[#############[#############[##############[#######
#.........................................................#
#.^.....................................................v.#
#.........................................................#
########[##############/############[###############[######
#..............#.............#...............#............#
#..............#.............#...............#............#
#..............#.............#.............g.#............#
#...........gg.#.............#..............g#....g..g....#
#######################.###################################
#.........................................................#
#.........................................................#
#....................................................@.v..#
#.........................................................#
###########################################################
Bacher                         HP:  10/  10
```

2. Second floor: proceed through the middle door in the lower set of rooms. There is a hidden door in the wall. Proceed through the hidden door and down the lower stairs.



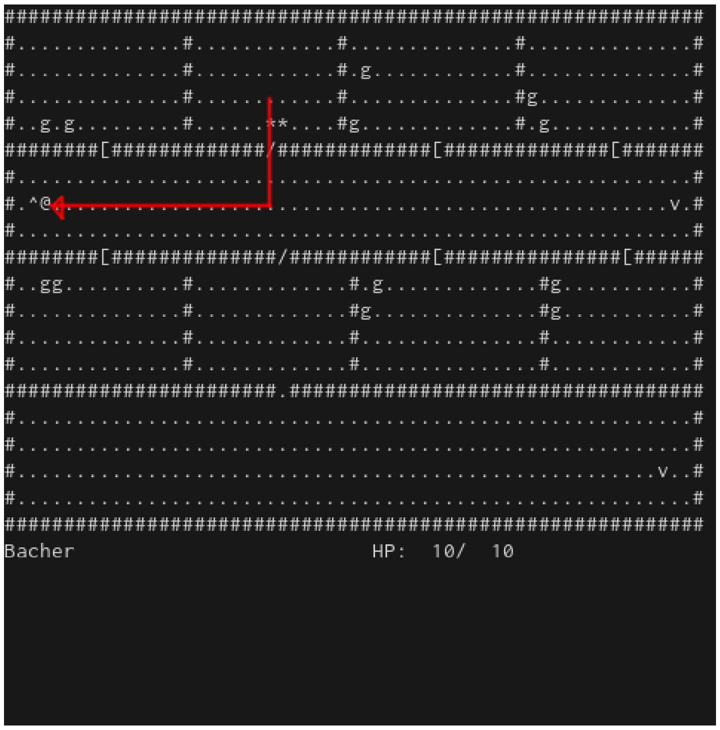
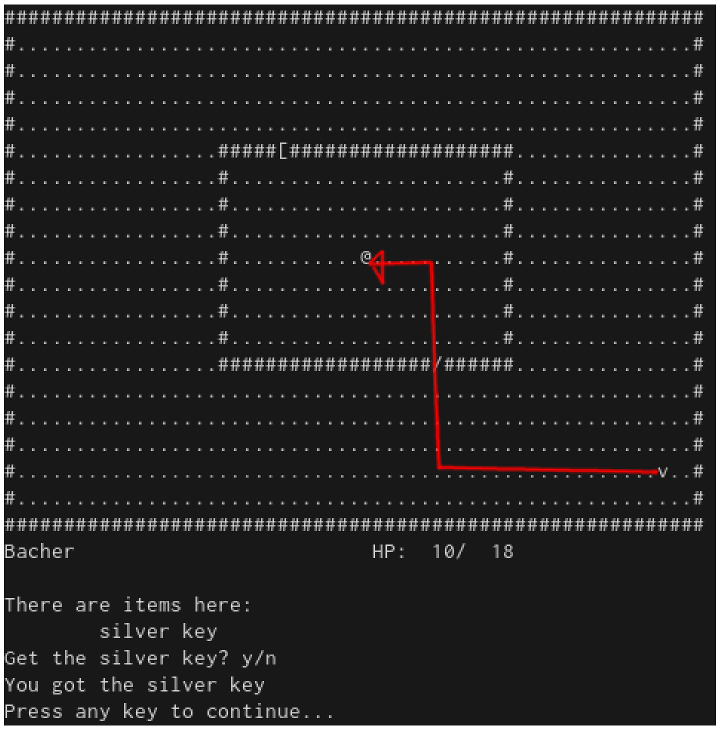
```
#################
#...............#
#...............#
#.......^.......#
#..........@....#
#...............#
#################
Bacher                          HP:  10/  10

There are items here:
        great sword
        plate mail armor
Get the great sword? y/n
You got the great sword
Get the plate mail armor? y/n
You got the plate mail armor
Press any key to continue...
```

3. Hidden room: in this room, there is a *well-dressed goblin*, a special goblin carrying absurdly good items that will make you nearly invincible and that will make testing the game significantly easier. Take and equip these items if you want, then return to the floor above.



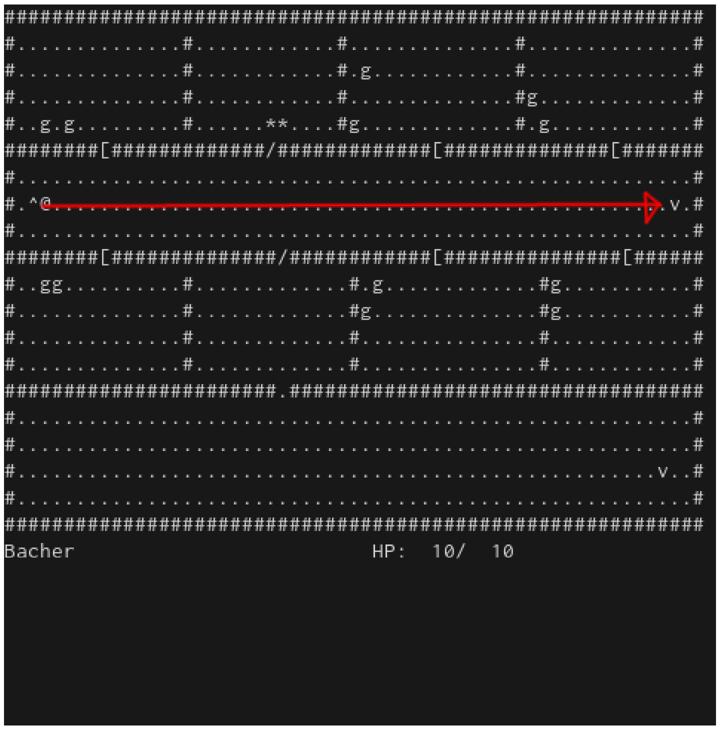
```
###########################################################
#.............g#............#..............#..............#
#..............#............#.g............#..............#
#.............g#.......@....#..............#g.............#
#..............#......**....#g.............#.g............#
########[#############/#############[##############[#######
#.........................................................#
#.^.....................................................v.#
#.........................................................#
########[##############/############[###############[######
#............gg#.............#.g.............#g..g........#
#..............#.............#g..............#............#
#..............#.............#...............#............#
#..............#.............#...............#............#
#######################.###################################
#.........................................................#
#.........................................................#
#......................................................v..#
#.........................................................#
###########################################################
Bacher                          HP:  10/  10

There are items here:
        brass key
Get the brass key? y/n
You got the brass key
Press any key to continue...
```

4. Second Floor: From the downstair, proceed to the indicated room and obtain the brass.

	5. Return to the first floor.
	6. Open the locked door, kill the skeletons, take the silver key and return to the second floor.

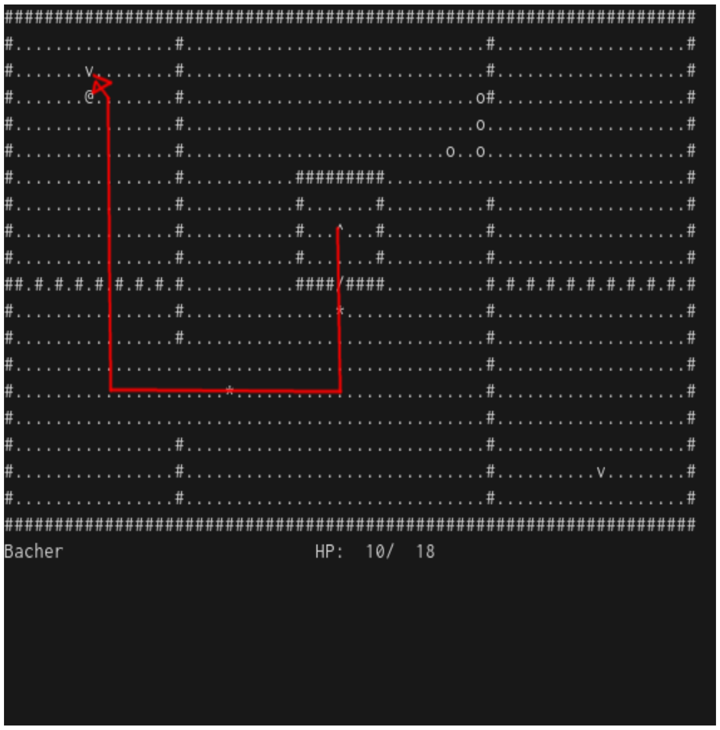
```
###########################################################
#..............#............#..............#..............#
#..............#............#.g............#..............#
#..............#............#..............#g.............#
#..g.g.........#......**....#g.............#.g............#
########[#############/#############[##############[#######
#.........................................................#
#.^@....................................................v.#
#.........................................................#
########[##############/############[###############[######
#..gg..........#.............#.g.............#g...........#
#..............#.............#g..............#g...........#
#..............#.............#...............#............#
#..............#.............#...............#............#
#######################.###################################
#.........................................................#
#.........................................................#
#......................................................v..#
#.........................................................#
###########################################################
Bacher                         HP:  10/  10
```

```
###########################################################
#.........................................................#
#.........................................................#
#.........................................................#
#.........................................................#
#.................#####[###################...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................#...........@...........#...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................#.......................#...............#
#.................##################/######...............#
#.........................................................#
#.........................................................#
#.........................................................#
#......................................................v..#
#.........................................................#
###########################################################
Bacher                         HP:  10/  18

There are items here:
        silver key
Get the silver key? y/n
You got the silver key
Press any key to continue...
```



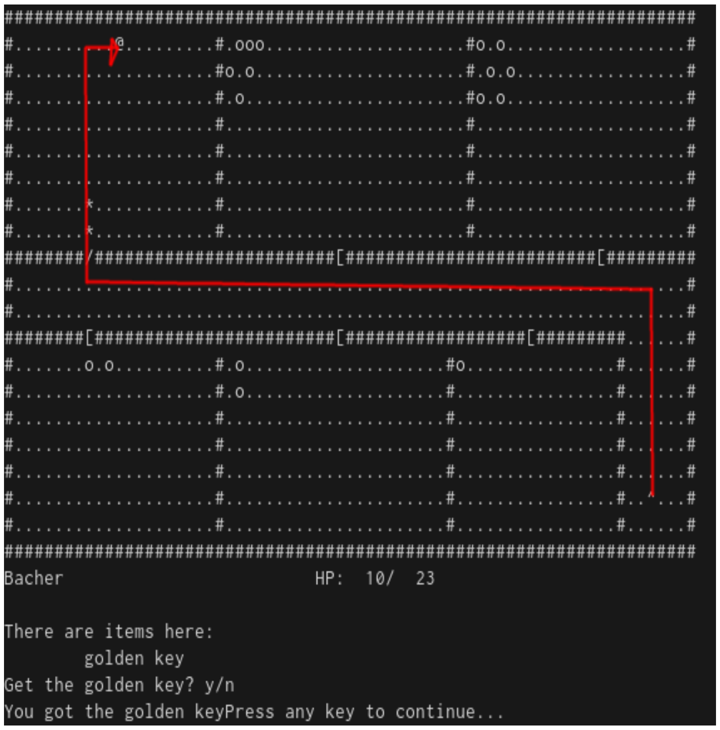
```
###########################################################
#..............#............#..............#..............#
#..............#............#.g............#..............#
#..............#............#..............#g.............#
#..g.g.........#......**....#g.............#.g............#
########[#############/#############[##############[#######
#.........................................................#
#.^@....................................................v.#
#.........................................................#
########[##############/############[###############[######
#..gg..........#.............#.g.............#g...........#
#..............#.............#g..............#g...........#
#..............#.............#...............#............#
#..............#.............#...............#............#
#######################.###################################
#.........................................................#
#.........................................................#
#......................................................v..#
#.........................................................#
###########################################################
Bacher                         HP:  10/  10
```

7. Proceed one floor down.



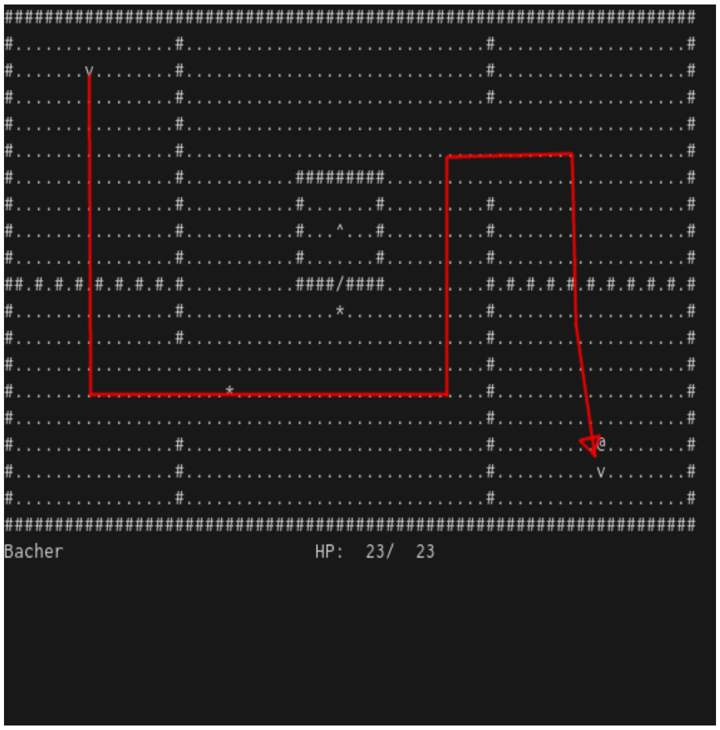
```
#####################################################################
#................#..............................#...................#
#.......v........#..............................#...................#
#.......@........#.............................o#...................#
#................#.............................o....................#
#................#..........................o..o....................#
#................#...........#########..............................#
#................#...........#.......#..........#...................#
#................#...........#...^...#..........#...................#
#................#...........#.......#..........#...................#
##.#.#.#.#.#.#.#.#...........####/####..........#.#.#.#.#.#.#.#.#.#.#
#................#...............*..............#...................#
#................#..............................#...................#
#...............................................#...................#
#.....................*.........................#...................#
#...............................................#...................#
#................#..............................#...................#
#................#..............................#..........v........#
#................#..............................#...................#
#####################################################################
Bacher                         HP:  10/  18
```

8. Optionally, kill the goblins and orcs on this floor, then take the staircase in the upper left.



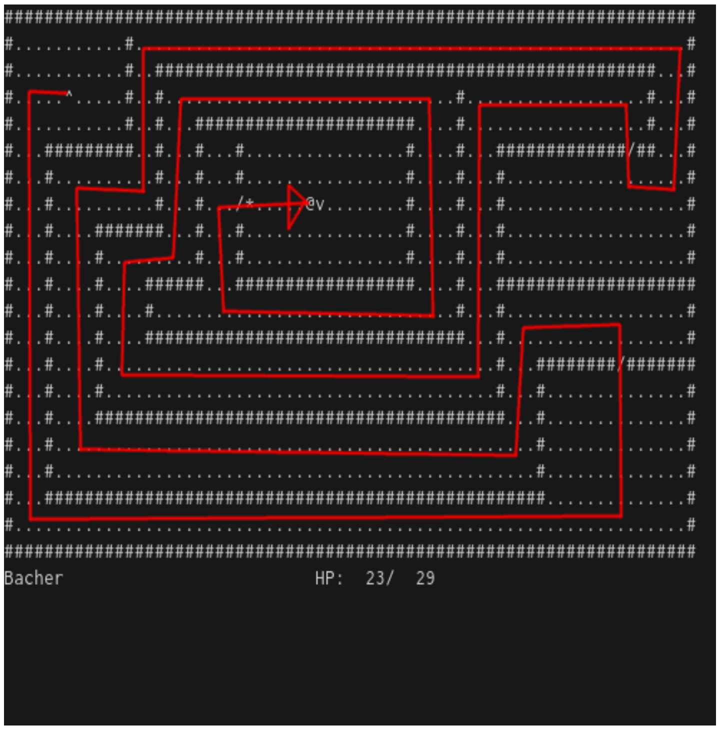
```
######################################################################
#..........@.........#.ooo....................#o.o...................#
#....................#o.o.....................#.o.o..................#
#....................#.o......................#o.o...................#
#....................#........................#......................#
#....................#........................#......................#
#....................#........................#......................#
#.......*............#........................#......................#
#.......*............#........................#......................#
########/########################[#########################[##########
#....................................................................#
#....................................................................#
########[########################[##################[#########.......#
#.......o.o..........#.o....................#o...............#.......#
#....................#.o....................#................#.......#
#....................#......................#................#.......#
#....................#......................#................#.......#
#....................#......................#................#.......#
#....................#......................#................#...^...#
#....................#......................#................#.......#
######################################################################
Bacher                         HP:  10/  23

There are items here:
        golden key
Get the golden key? y/n
You got the golden keyPress any key to continue...
```

9. Proceed to the room at the far corner and get the golden key, then return to the floor above.



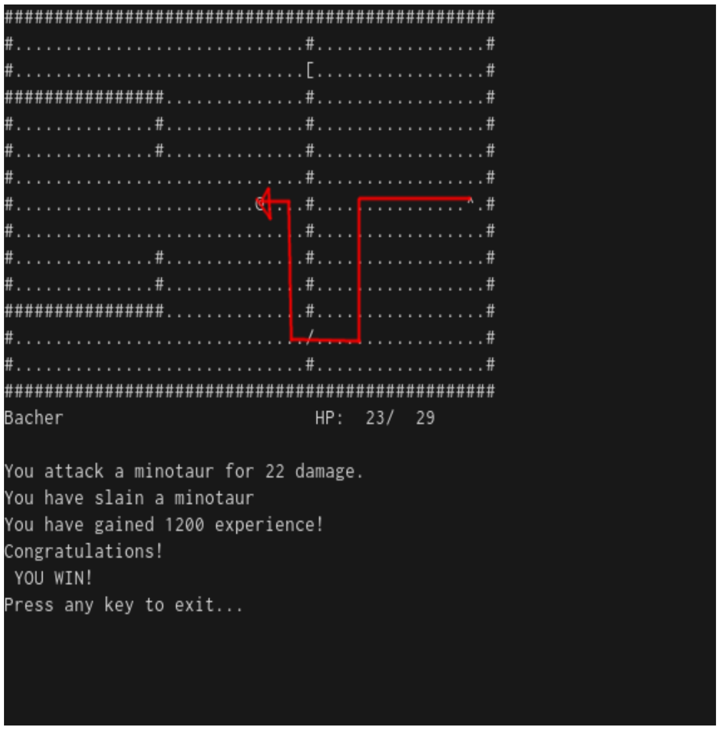
```
######################################################################
#................#...............................#...................#
#.......v........#...............................#...................#
#................#...............................#...................#
#................#...................................................#
#................#...................................................#
#................#...........#########...............................#
#................#...........#.......#...........#...................#
#................#...........#...^...#...........#...................#
#................#...........#.......#...........#...................#
##.#.#.#.#.#.#.#.#...........####/####...........#.#.#.#.#.#.#.#.#.#.#
#................#...............*...............#...................#
#................#...............................#...................#
#................................................#...................#
#.....................*..........................#...................#
#................................................#...................#
#................#...............................#..........@........#
#................#...............................#..........v........#
#................#...............................#...................#
######################################################################
Bacher                         HP:  23/  23
```

10. Travel across the room and take the stairs down in the bottom right corner of this floor.



```
#####################################################################
#...........#.......................................................#
#...........#..##################################################...#
#.....^.....#..#.............................#..................#...#
#...........#..#...######################....#..................#...#
#...#########..#...#...#................#....#...#############/##...#
#...#..........#...#...#................#....#...#..................#
#...#..........#...#.../*.....@v........#....#...#..................#
#...#....#######...#...#................#....#...#..................#
#...#....#.........#...#................#....#...#..................#
#...#....#....######...##################....#...####################
#...#....#....#..............................#...#..................#
#...#....#....################################...#..................#
#...#....#.......................................#...########/#######
#...#....#.......................................#...#..............#
#...#....#########################################...#..............#
#...#................................................#..............#
#...#................................................#..............#
#...##################################################..............#
#...................................................................#
#####################################################################
Bacher                         HP:  23/  29
```

11. Travel around the spiral labyrinth, killing the skeletons in this room, then open the locked door with the golden key and take the stairs down.



```
#################################################
#.............................#.................#
#.............................[.................#
################..............#.................#
#..............#..............#.................#
#..............#..............#.................#
#.............................#.................#
#........................@....#...............^.#
#.............................#.................#
#..............#..............#.................#
#..............#..............#.................#
################..............#.................#
#............................./.................#
#.............................#.................#
#################################################
Bacher                         HP:  23/  29

You attack a minotaur for 22 damage.
You have slain a minotaur
You have gained 1200 experience!
Congratulations!
 YOU WIN!
Press any key to exit...
```

12. Kill the bugbears in the antechamber, then proceed through the door and kill the Minotaur.

That’s it. You win!

Appendix A. Global Constant Table

Name	Data Type	Value	Defined
PC_RENDER_C	char	‘@’	Character.hpp
STARTING_COORD	Coord	Coord(4,4)	Character.hpp
MAX_DAYS	int	5	Game.hpp
MAP_PATH_ROOT	std::string	‟gamedata/maps/’	Game.hpp
ITEM_TBL_PATH	std::string	‘gamedata/items/items.tbl’	Game.hpp
WPN_TBL_PATH	std::string	‘gamedata/items/weapons.tbl’	Game.hpp
ARMR_TBL_PATH	std::string	‘gamedata/items/armor.tbl’	Game.hpp
LOGFILE_PATH	std::string	‘gamedata/log’	Game.hpp
MOB_TBL	std::string	‘gamedata/mobs/mobs.tbl’	Game.hpp
MOB_LOOT_DIR	std::string	‘gamedata/mobs/loot/’	Game.hpp
STARTING_MAP	std::string	‘floor001’	Game.hpp
STARTING_WPN	std::string	‘l_sword’	Game.hpp
STARTING_AMR	std::string	‘starter_scale_mail’	Game.hpp
QUEST_TARGET_ID	std::string	‘mino01’	Game.hpp
QUEST_TARGET_FLOOR	std::string	‘floor007’	Game.hpp
QUEST_TARGET_COORD	Coord	Coord(4,7)	Game.hpp
OPEN_DOOR_C	char	‘/’	Space.hpp
CLOSED_DOOR_C	char	‘[’	Space.hpp
EMPTY_SPACE_C	char	‘.’	Space.hpp
ITEM_SPACE_C	char	‘*’	Space.hpp
WALL_C	char	‘#’	Space.hpp
UP_STAIR_C	char	‘^’	Space.hpp
DOWN_STAIR_C	char	‘v’	Space.hpp
HIDDEN_DOOR_C	char	‘%’	Space.hpp