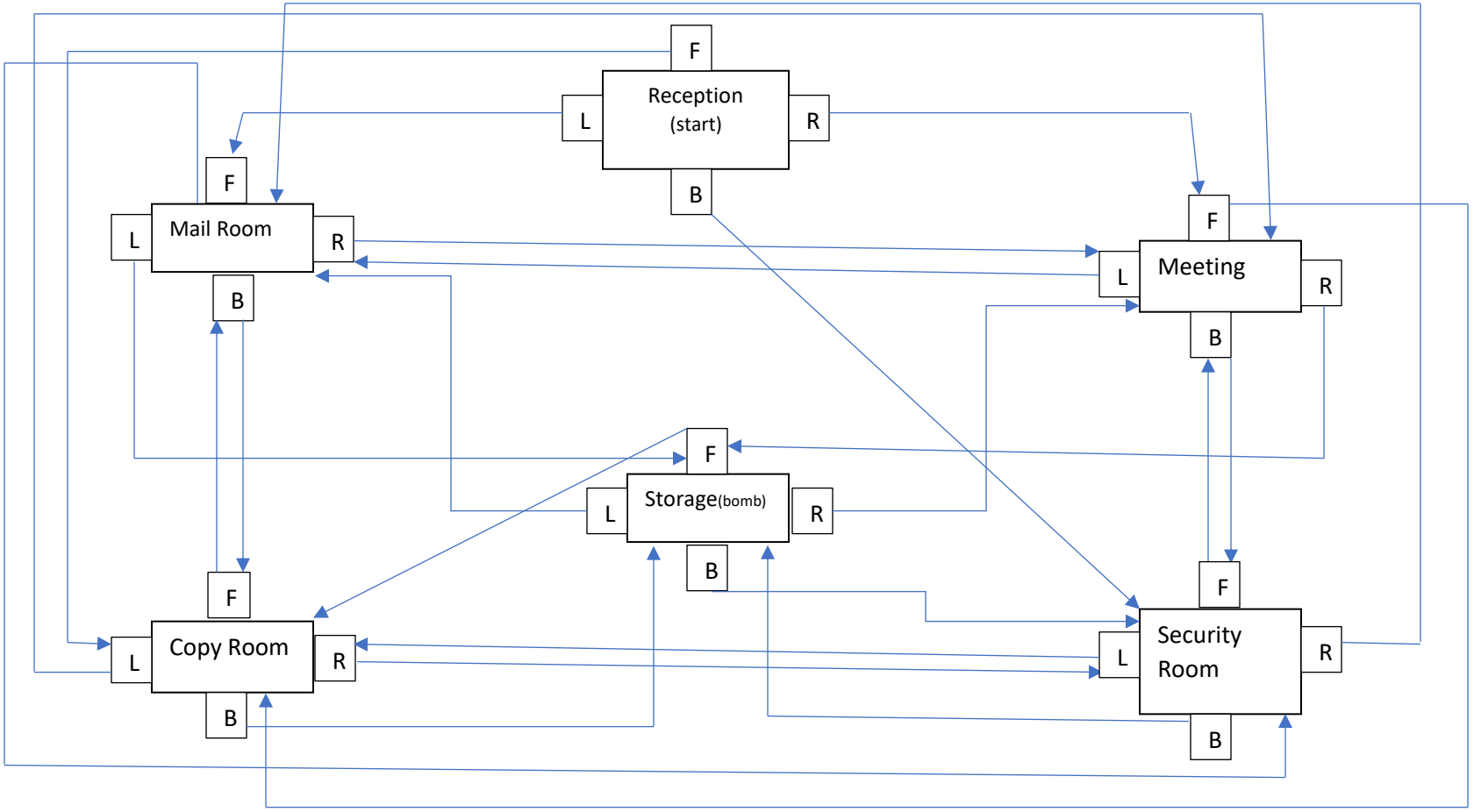
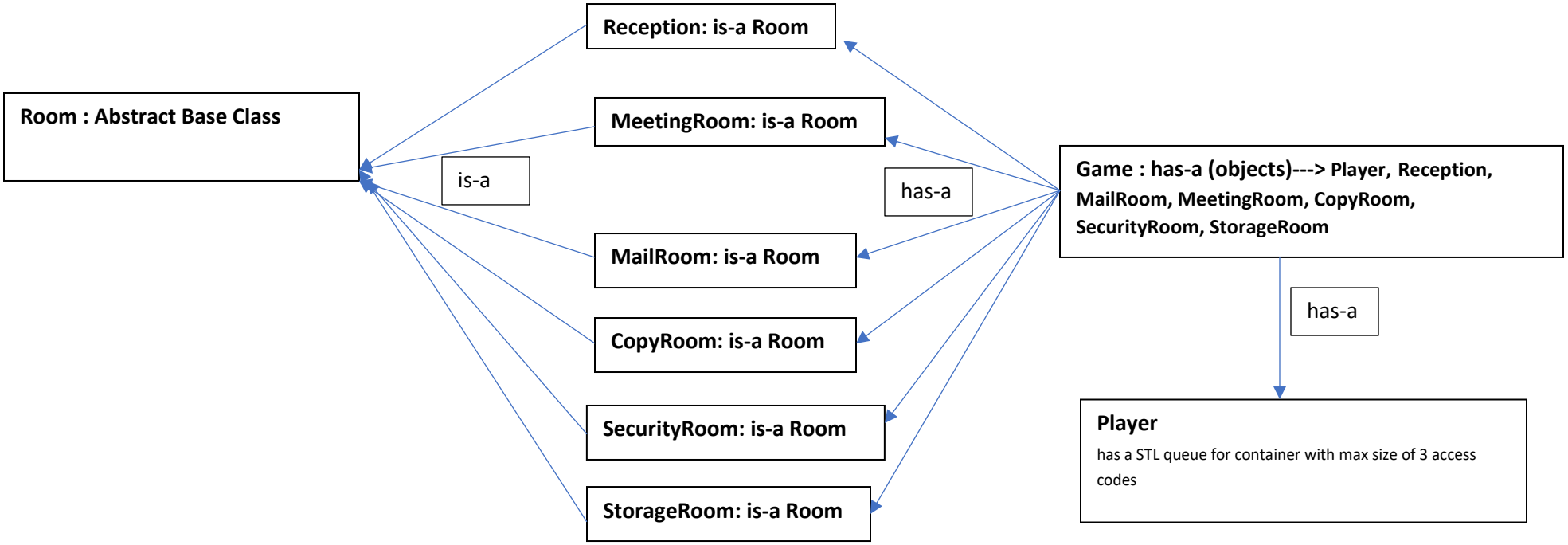
Robert Boespflug
Final Project Testing/Reflection Document

Room Pointer Diagram

6 rooms, 4 pointers used in every room to link to another room.
User starts in start room.
From there can go to any game room, but not the bomb room.
From a game room, can go to any other game room or bomb room, but not back to start room.



Original Design



Updated Design

Room : Abstract Base Class

public:

```
std::string name;
Room * right = nullptr;
Room * left = nullptr;
Room * forward = nullptr;
Room * back = nullptr;     //NO UNUSED POINTERS
Room();
virtual ~Room();
void printName();
void setRightPtr(Room * nextRoom); /*FUNCTIONS USES DIRECTION
void setLeftPtr(Room * nextRoom);  PTR'S  SET ABOVE TO MOVE
void setForwardPtr(Room * nextRoom); PLAYER AROUND THE ROOMS*/
void setBackPtr(Room * nextRoom);
```

Game : has-a ---> StartRoom, GuessRoom, BombRoom, Player

private:

```
ReceptionRoom * reception = new ReceptionRoom;
MailRoom * mailRoom = new MailRoom;
MeetingRoom * meetingRoom = new MeetingRoom;
CopyRoom * copyRoom = new CopyRoom;
SecurityRoom * securityRoom = new SecurityRoom;
StorageRoom * storageRoom = new StorageRoom;
Player player;
Room * playerLocation;
int nextPLayerMove;
std::chrono::time_point<system_clock> begin;
std::chrono::time_point<system_clock> end;
int timeAvailable = 600;
bool userExit = false;
```

public:

```
Game();
~Game();
void startGame();
void playGame();
void displayPrompt();
string getAccessCode();
void showAccessCodeCount();
bool bombNotDefused();
void tryDefusingBomb();
int elapsedTime();
int timeLeft();
bool time_is_left();
void playReceptionRoom();
void playMailRoom();
void playMeetingRoom();
void playCopyRoom();
```

ReceptionRoom : is-a Room

private:

```
bool lightSwitch = false;
const int MENU_CHOICE_LIGHTS = 1;
const int MENU_CHOICE_DIRECTIONS = 2;
const int MENU_CHOICE_MEETING = 3;
const int MENU_CHOICE_MAIL = 4;
const int MENU_CHOICE_COPY = 5;
const int MENU_CHOICE_SECURITY = 6;
const int MENU_CHOICE_EXIT = 7;
```

public:

```
ReceptionRoom();
virtual int menu() override;
virtual void specialAction() override; //ROOM STATUS CHANGED HERE
void readDirections();                          ->turns on lights
```

MeetingRoom: is-a Room

private:

```
int numMoves = 0;
char tictactoeBoard[3][3];
char playerTurn;
char computerTurn;
bool gameOver = false;
int row = 0;
int column = 0;
int computerRow = 0;
int computerColumn = 0;
enum Gamestates { X_WON, O_WON, DRAW, UNFINISHED };
bool makeMove(int row, int column, char playerTurn);
void checkGameState();
int gameState();
int checkRows();
int checkColumns();
int checkDiagonals();
int getRandomInt();
int checkDraw();
void pickFirstTurn();
void printBoard();
void getPlayerMove(char playerTurn);
void getComputerMove(char computerTurn);
void tictactoeMenu(char playerTurn);
void createBoard();
const int THREE_IN_ROW = 3;
const int MENU_CHOICE_LEVER = 1;
const int MENU_CHOICE_COPY = 2;
const int MENU_CHOICE_SECURITY = 3;
const int MENU_CHOICE_MAIL = 4;
const int MENU_CHOICE_STORAGE = 5;
```

public:

```
bool leverPulled = false;
MeetingRoom();
void playTicTacToe();
virtual int menu() override;
virtual void specialAction() override;
```

CopyRoom : is-a Room

private:

```
void rockPaperScissors();
void assignUserChoice(int menuChoice);
void resetGame();
```

public:

```
bool leverPulled = false;
CopyRoom();
virtual int menu() override;
virtual void specialAction() override; //SPECIAL ACTION
```

Player

```
private:
        Room * location;
public:
        queue<string> notebook;
        void changeLocation(Room *);
        Room * getCurrentLocation();                    //TRACKS PLAYER LOCATION
        void addAccessCode(std::string);
        void removeAccessCode();
        void displayAccessCode();
        string getAccessCode();
```

StorageRoom : is-a Room

```
public:
        bool bombDefused = false;       //ROOM STATUS CHANGED HERE
        bool bombBlownUp = false;       //OR HERE
        string accessCode = "xyz";
        string userInput;
        virtual int menu() override;
        void enterAccessCode(string input);
        virtual void specialAction() override;          //BLOWS UP BOMB
        void defuseBomb();                              //defuse Bomb
```

More Below

Problems/Solutions:

1. I needed a Goal and theme for my game.
   -> Terrorist have taken over the player's office building, and the user must move the player around the rooms of the office building and gather access codes to defuse the bomb.

2. I needed to create at least six "Spaces".
   -> I used classes to represent the spaces as rooms in my game. I have an abstract 'Room' class and six derived 'Room' classes: Reception room, meeting room, mail room, copy room, security room, and the storage room.

3. I wanted to use **ALL** 4 four pointers in every room.
   -> See the graph on first page that represents how the rooms are linked together. Basically, the starting room links to the four game rooms, that all link to the bomb room and themselves. The bomb room links to the four game rooms.

4. Each room needed a special action.
   -> I used a pure virtual function specialAction() in the abstract room class. Every office room has this function.

5. I needed a way to keep track of what room the player was in.
   -> I have the getCurrentLocation() function in the player class that returns that room the player is in.

6. I needed to have a time limit or some way to stop the game for playing endlessly.
   -> I have a time limit of 10 minutes, if the user has not defused the bomb by then, the bomb explodes and the game is over. I used the "system_clock" features of c++11 for timing code.

7. I needed to interact with parts of each room
   -> The reception room turns on the lights, the four game rooms have a lever that needs to be activated to unlock the doors, and the storage room has a bomb that needs to be defused and that will blow up and end the game if not defused.

8. I needed a container to gather items in the rooms.
   -> I have a STL queue that is represented in the game as a "notebook" the user uses to write down access codes.

9. The container needed some sort of maximum value.
   -> The queue cannot have more than three access codes at a time.

10. I needed a 'player' for the user to control.
    -> There is a 'Player' class and the Game class has a *player* object.

11. I needed a central class that ran the game where I had access to both the “Player” member attributes and the “Room” (or derived Room) member attributes.
    -> I have a Game class that has the six office rooms, and a player to play the game. (Objects). From here both there is access to both the Room or derived Room members and the Player members.

12. I needed to eliminate free-form input to cut-down on spelling errors.
    -> I have menu options at almost every option, and every input is an integer, and is validated for an integer and in between a correct range.

Testing Plan:

My main plan for testing is to run through the game, anticipating what the expected output should be at every step. Then, I will test for more specific things like the time limit, the bomb exploding, the bomb being defused, etc.

Along the way, I am making note of if the movement around the rooms using the pointers of every room to see if they lead to the right place. Then, test the games to see if they play correctly, and finally, seeing if the interaction of player to room works for every room and that the game and the objective plays correctly.

More Below

Testing Results:

Printed to Screen	Actual Input	Expected Output	Actual Output	Correct
WELCOME!! 1) Play Game 0) Exit ------>	1	Displays Starting Game Prompt	Welcome to the game!!! … ---> Press any key to start time:	Yes
**TIME LEFT: 300 seconds! --> You're in the RECEPTION room! 1) TOGGLE breaker that controls the LIGHTS in office building 7) END GAME<br-------->	1	Turn on lights, display the rest of the reception room menu, time should go down some from 300	**TIME LEFT: 244 seconds! --> You're in the RECEPTION room! 1) TOGGLE breaker that controls the LIGHTS in office building 2) READ piece of paper in pocket 3) Go to the MEETING room 4) Go to the MAIL room 5) Go to the COPY room 6) Go to the SECURITY room 7) END GAME<br-------->	Yes
**TIME LEFT: 244 seconds! --> You're in the RECEPTION room! 1) TOGGLE breaker that controls the LIGHTS in office building 2) READ piece of paper in pocket 3) Go to the MEETING room 4) Go to the MAIL room 5) Go to the COPY room 6) Go to the SECURITY room 7) END GAME<br-------->	3	Go to the meeting room, time reduced, 0 access codes in notebook	**TIME LEFT: 124 seconds! **Access Codes in notebook: 0 ----->MEETING Room 1) Activate lever 2) Go to COPY room 3) Go to SECURITY room 4) Go to MAIL room 5) Go to STORAGE room<br-------->	Yes

**TIME LEFT: 124 seconds! **Access Codes in notebook: 0 ----->MEETING Room 1) Activate lever 2) Go to COPY room 3) Go to SECURITY room 4) Go to MAIL room 5) Go to STORAGE room -------->	1	Start tictactoe game.	Pick who goes first 1: X 2: O --->	Yes
---> X WON! YOU ARE ACTIVATING THE LEVER! ---> Doors UNLOCKING! ---> Access code is put in notepad! **TIME LEFT: 291 seconds! **Access Codes in notebook: 1 ----->MEETING Room 1) Activate lever 2) Go to COPY room 3) Go to SECURITY room 4) Go to MAIL room 5) Go to STORAGE room -------->	2	Go to copy room, time reduced, one item in notebook	**TIME LEFT: 183 seconds! **Access Codes in notebook: 1 ----->COPY Room 1) Activate lever 2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room -------->	Yes
**TIME LEFT: 183 seconds! **Access Codes in notebook: 1	2	Deny access to security room, go back to menu, time reduced	----> Door locked! Pull lever to open!! **TIME LEFT: 117 seconds!	Yes

----->COPY Room 1) Activate lever 2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room<br-------->			**Access Codes in notebook: 1 ----->COPY Room 1) Activate lever 2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room<br-------->	
**TIME LEFT: 117 seconds! **Access Codes in notebook: 1 ----->COPY Room 1) Activate lever 2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room<br-------->	1	Start rock, paper, scissors game	----> Door locked! Pull lever to open!! **TIME LEFT: 117 seconds! **Access Codes in notebook: 1 ----->COPY Room 1) Activate lever 2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room<br-------->	
********** ----> ROUND: 7 --> User pick: Paper --> Computer pick: Scissors User wins: 1 Computer wins: 3 ********** ---> Computer Won, Try again!!	2	Deny access to security room, bomb exploded, time limit reached, user prompted to go again	----> Door locked! Pull lever to open!! ********** YOU TOOK TOO LONG!! BOMB EXPLODED! --> Elapsed time: 328 seconds! ---> Thank you for playing!! Would you like to go again? ---> Press 1 for yes and 0 for no --->	Yes
**TIME LEFT: 124 seconds! **Access Codes in notebook: 0 ----->MEETING Room	3	Go to mail room	**TIME LEFT: 597 seconds! **Access Codes in notebook: 0 ----->MAIL Room 1) Pull lever	Yes

1) Activate lever 2) Go to COPY room 3) Go to SECURITY room 4) Go to MAIL room 5) Go to STORAGE room -------->			2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room (Bomb) -------->	
**TIME LEFT: 598 seconds! --> You're in the RECEPTION room! 1) TOGGLE breaker that controls the LIGHTS in office building 2) READ piece of paper in pocket (DIRECTIONS) 3) Go to the MEETING room 4) Go to the MAIL room 5) Go to the COPY room 6) Go to the SECURITY room 7) END GAME<br-------->	4	Access Blocked, go back to Mail Room Menu	----> Door locked! Pull lever to unlock!! **TIME LEFT: 528 seconds! **Access Codes in notebook: 0 ----->Mail Room 1) Pull lever 2) Go to SECURITY room 3) Go to MEETING room 4) Go to MAIL room 5) Go to STORAGE room (Bomb) -------->	Yes
**TIME LEFT: 599 seconds! **Access Codes in notebook: 0 ----->MEETING Room 1) Pull lever 2) Go to COPY room 3) Go to SECURITY room 4) Go to MAIL room 5) Go to STORAGE room -------->	1	Start tic-tac-toe game	Pick who goes first 1: X 2: O --->	Yes
**TIME LEFT: 588 seconds! **Access Codes in notebook: 0	1	Start guessing game	Guess an INTEGER between 1-20 (binary search?): 5 guesses left!! --->	Yes

-----> MAIL Room 1) Pull Lever 2) Go to MEETING room 3) Go to SECURITY room 4) Open door to the COPY room 5) Take stairs to STORAGE room (Bomb) -------->				
What is being tested		**How**	**Outcome**	**Success?**
Time limit		Playing longer than time limit of 8 seconds	********** YOU TOOK TOO LONG!! BOMB EXPLODED! YOUR OFFICE BUILDING COLLAPSES! --> Elapsed time: 8 seconds! ---> Thank you for playing!! Would you like to go again? ---> Press 1 for yes and 0 for no	Yes
Bomb being defused with invalid access code		Getting access codes and getting to storage room to defuse bomb.	---> ENTERING ACCESS CODE... ---> INVALID ACCESS CODE!! ACCESS CODE REMOVED!!	Yes
Bomb being defused with *valid* access code		Getting access codes and getting to storage room to defuse bomb.	---> ENTERING ACCESS CODE... ********** ---> You defused the bomb!! YOU ARE THE OFFICE **HERO**!! It took you: 22 seconds! You had: 578 seconds left! ---> Thank you for playing!! Would you like to go again? ---> Press 1 for yes and 0 for no	Yes

Testing Results

Everything seems to be working as expected. The player can move around the rooms nicely after he/she plays and wins the game in room after pulling the lever in that room. The interaction of player to rooms is working nicely, with the specialAction() of every room doing its job. The time limit on the game seems to be working correctly, with the bomb exploding if the user plays for longer than the time limit which can be accessed in GAME.HPP. The bomb is defusing only if the user plays and wins the game in the MAIL room, and comes to the STORAGE room and picks to defuse the bomb. The game correctly displays the time it took the player to defuse and how much time was left on the clock.