Unity を用いた 2DRPG 制作

安藤 幸輝　北里 健人
野中 駿佑

１．研究概要

Unity を用いたゲームの共同開発を通してゲーム制作の技術を磨くことを目標に 2DRPG を制作した。RPG の機能として敵との戦闘、セーブ・ロード、アイテムの管理や売買などのシステムを実装した。

２．研究の具体的内容

（１） ゲーム概要

今回制作したゲームは、仮想の神話「クトゥルフ神話」をモチーフにした。ストーリに沿って武器や装備を使い、敵を倒していく内容となっている。

プレイヤの移動は、一般的な WASD キーと矢印キーの両方を採用していている（図１）。

アイテムの使用やセーブなどが確認できるシステムの UI は、図２のように視覚的にわかるシンプルデザインかつ操作しやすいようなものにしている。

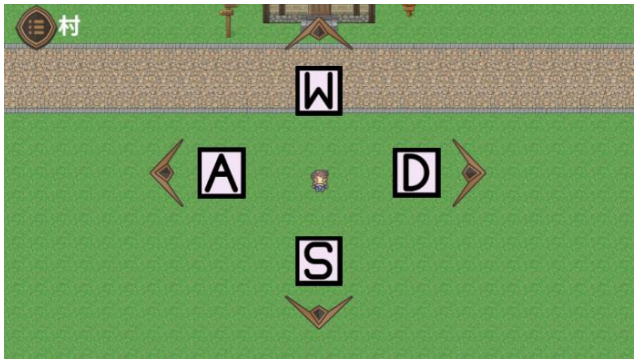

図１　プレイヤ操作

図２　システム UI

ア　作成スケジュール

このゲームを作るにあたって、機能ごとに分類したスケジュールを表１のように計画した。機能ごとに担当を分けて作業をしたため予定通りの機能を実装することができた。

表１　作成スケジュール

４月	プレイヤ、ショップの機能実装
５月	システム UI の実装
６月	マップ制作、アイテム機能実装
７月	戦闘システム実装、ストーリ制作
８月	戦闘システム改善、ストーリ完成
９月	クエスト機能の実装
10 月	アイテム機能、バグの改善
11 月	無限ダンジョン、鍛冶システム
12 月	資料作成
１月	課題研究発表準備

イ　開発環境

ゲーム制作で使用した開発環境は次のとおりである。

・Unity2022.3.8f1
・Microsoft Visual Studio Community 2022 (64 ビット) - CurrentVersion 17.11.3
・SourceTree version3.4.18
・GitHub

ウ　制作の役割分担

一般的にゲームを作るために、音楽・デザイン・プログラミングなどが必要になってくる。プログラミングは、機能ごとに分担して、絵や音楽はフリー・有料素材を用いて制作した。分担した内容は表２のようになっている。

表２　機能ごとの役割分担

安藤	戦闘、 ストーリ構成、敵の設定 ゲーム考案
北里	マップ、セーブ・ロード 無限ダンジョン、開発環境構築 SE・BGM システム
野中	システム UI、ショップ・アイテム システム、戦闘 UI・エフェクト プレイヤの基本操作、クエスト

（２）　ゲームの機能

ゲームを構成する主なシステムとして、アイテムボックス・戦闘システム・セーブロードなどがある。

ア　アイテムボックス

アイテムボックスでは、プレイヤが獲得したアイテムを使用・装備することができる。アイテムの種類には、消費アイテム・武器・装備の３種類があり、武器と装備はプレイヤが既に装備を付けていた場合、現在のアイテムと入れ替えるようにした。また、アイテムボックスの利便性を上げるためにアイテムの種類ごとに絞り込みやソートの機能及び、アイテムアイコンにマウスカーソルを合わせた時にそのアイテムの情報をテキストで表示する機能の実装を行った(図３)。

プレイヤ及びゲームシステムとして意図していない処理が行われないように図４のような処理手順でアイテムの使用を行っている。

図３　アイテムボックス画面

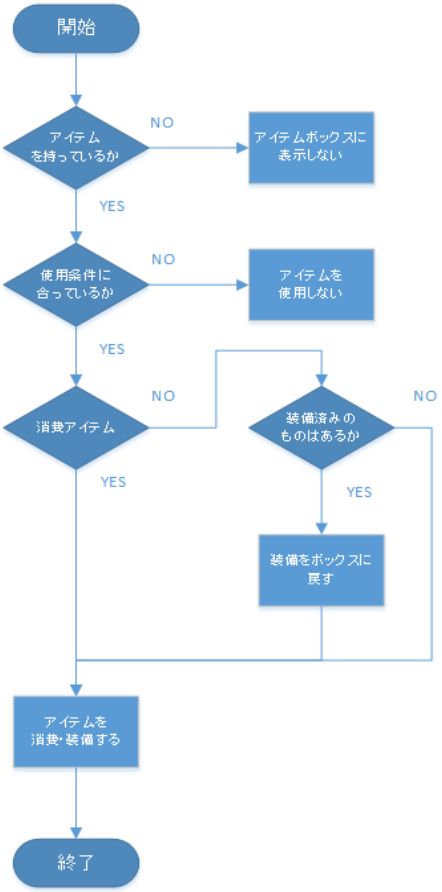

図４　アイテム使用時の処理

アイテムボックスの内部処理は、Unity の機能である ScriptableObject を利用している。このシステムを使うことで、アイテムの ID・名前・画像などの型の異なるものをまとめて管理することができる。メリットとして、入力されている値が実行後も保存されることと、保存されている値を視覚的に確認・変更ができる点である(図５)。

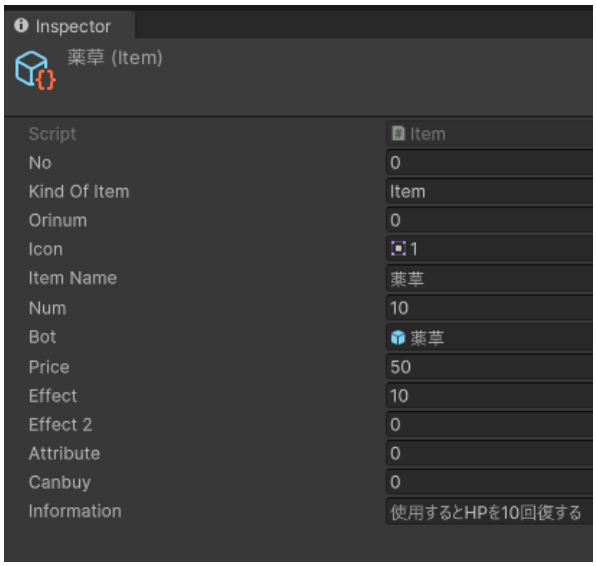

図５　アイテムごとの情報

イ　戦闘

戦闘はモンスタと戦い、経験値やアイテムを手に入れることができる機能である。武器や魔法を使っての攻撃や、アイテムの使用、逃亡といった基本的な機能を実装した。

(a)　武器について

武器の攻撃はプレイヤの攻撃値を参照し、ランダムに攻撃値の倍率が変わるようになっている。ほかにもプレイヤの攻撃力とモンスタの防御力の差を参照し、ダメージが出すぎないようにバランスをとっている。

・武器の特徴

武器ごとに特徴を持たせており、プレイヤが装備している武器種の識別番号を参照し、機能を実装している。

槍という武器種は連続攻撃ができたり、弓という武器種は敵の攻撃の命中率を下げるという特徴がある。

・武器の特殊能力

武器には特定の武器にだけ追加効果を実装しており、敵のレアドロップだったり、マップにある宝箱に入っていたりとレアな武器として実装している。図６のマジックソードという武器には、戦闘中に最後に使用した魔法の属性になるという効果があり、図７のウィンドスピアには攻撃回数が素早さのステータス÷10 の値だけ増えるという効果がある。

図６　マジックソード

図７　ウィンドスピア

(b) 属性相性

機能として属性相性機能を実装した。実装した属性は火、水、土、風、神話である。それぞれ属性には図８のような相性があり、有利の場合はダメージが２倍、不利の場合はダメージが半減と戦略に幅を持たせている。

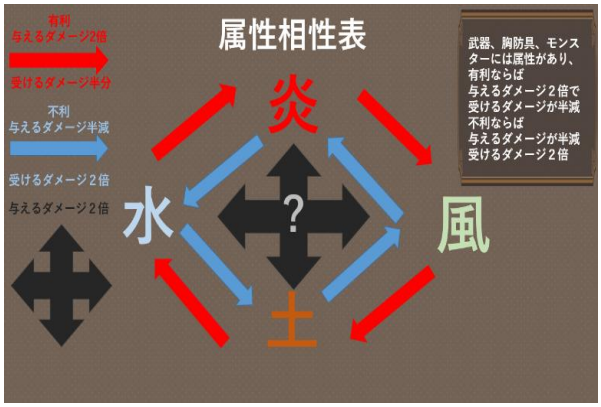

図８　属性相性

(c)　魔法機能

属性機能にもかかわってくるのが魔法機能である。魔法にも属性を持たせており、敵の弱点属性をつきやすくなっている。

・魔法強化

魔法には使えば使うほど攻撃力が上がる機能を実装しており、敵が強くなっても対応できるようになっている。

・魔法のクラス

魔法にはクラスがあり、１つの属性につき４種類の魔法を実装している。

魔法は ScriptableObject という Unity の機能を使って管理しており(図９)、スクリプト上で呼び出しやすくなっている。

図９　魔法管理画面

・魔法の命名について

魔法には名前に少しこだわっている。高いクラスの魔法には精霊の名前や、神話の神の名前を使っている。例を挙げると火の最高位魔法には人々に火を与えたギリシャ神話の神、プロメテウス。水の最高位魔法にはメソポタミア神話における原初の海の女神、ティアマト。風の最高位魔法には伊弉諾とイザナミの子であり、９つの首を持つ怪物ヤマタノオロチを打ち倒し、草薙剣を持ち帰ったとされるスサノオノミコト。土の最高位魔法にはアステカ神話の最高神、夜の空、黒き太陽、テスカトリポカといった神々の名を使用している。

そして、神話属性の魔法にはクトゥルフ神話に出てくる神の名を使用している。

(d)　魔法のエフェクト

魔法の演出をするために Unity の ParticleSystem を利用している。この機能を使うことで、数枚の画像でアニメーションを作りだすことができる(図 10)。設定を変更することで、アニメーションの出力方向や画像切り替えの滑らかさを調整することができる。

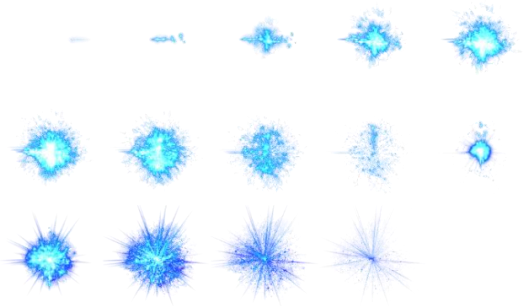

図 10　アニメーション用画像

(e)　戦闘 UI

戦闘をよりスムーズに行うことができるように UI を工夫した。アイテムボックスからアイテムの詳細を確認できるようにマウスカーソルが重なったアイテムの効果を表示している(図 11)。

図 11　アイテム情報の表示

また、アイテム詳細ボタンからプレイヤのステータスやバフ・デバフの有無を確認することができるようにもしている。

ウ　セーブ・ロード

ゲームデータの保存には JsonUtility という Unity の API を使用している。これは、様々な型の変数や配列を JSON ファイルとして保存することができるものである。これによりゲームを終了してもプレイヤのＨＰやストーリの進捗などのデータを保持し、続きから始められるようになっている。

エ　クエスト

クエストは、ギルドから受注することができる。討伐・発見・納品の３種類のクエストがあり、条件をクリアすることで薬草や剣などのアイテムをもらうことができる。クエストの受注画面では、引き受けていないクエストを確認することができる(図 12)。

また、クエストを一定数クリアすることで試練を受けることができ、クリアすることで自身のランクを上げることができる。ランクに応じてショップで買えるアイテムの質が向上する。

図 12　クエスト受注画面

オ　ステータスポイント

プレイヤはレベルが上がるごとにステータスポイントを獲得することができる。ステータスポイントを使用することで、ランダムで上昇するステータスに加えて任意のステータスを伸ばすことができる。ステータスの割り振り方を工夫することで戦闘を有利に進めることができる。

図 13　ステータスポイントの設定画面

カ　シーン移動

シーン移動は図 14 の行きと図 15 の帰りに分かれている。行きでは、現在のマップとプレイヤの位置を保存しておき、ドアに関連付けられた移動先のマップと位置の情報を読みこみ、シーン移動をしている。帰りでは、行きで保存したデータを使ってシーン移動をしている。このように行きと帰りで分けることによって、帰りの分のマップや位置のデータを用意しておく必要がなくなり、シーン移動のためのデータ量を半分にすることができる。

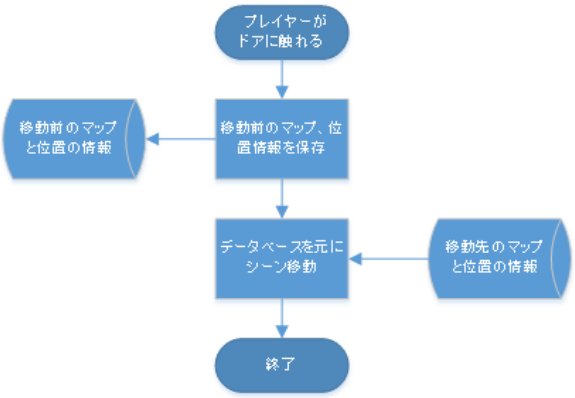

図 14　シーン移動の行き

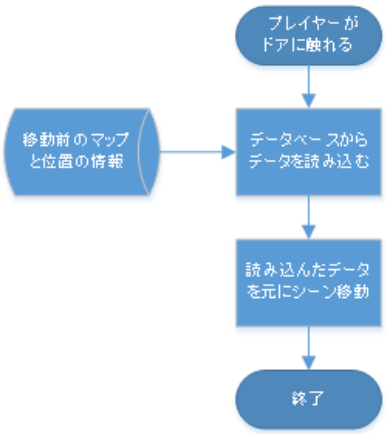

図 15　シーン移動の帰り

キ　教会

教会では、聖水の購入・教会への献金・祈りができる。

(a)　聖水

聖水は、特定のモンスタに対するダメージ上昇バフを得られる。

(b)　教会への献金

献金を行うとアイテムを得ることができ、教会からの信頼が上昇する。

(c)　祈り

神に祈ることで何か起こるかもしれない。

図 16　教会

ク　アイテムの売買

アイテムの売買を行うために、ショップを制作した。ショップでは、プレイヤのランクに応じたアイテムを購入することができる。また、戦闘で獲得したアイテムと使わなくなった武器などを売って通貨に変換することができる。

ケ　エンカウント

プレイヤが森・洞窟などで探索をしていると、ランダムで敵に遭遇することがある。プレイヤが歩いた歩数に応じて敵に遭遇するかどうかの抽選が行われる。

タイルマップの 1 マスを 1 歩とカウントするように移動距離を測って計算を行った(図 17)。

```
void lottary()
{
    if (walkcount % 10 == 0 && walkcount != 0 && walk == true)
    //１０歩ごとにエンカウントの抽選
    {
        encountcheck = true;
        Invoke("ram", 0.2f);
    }
}
0 個の参照
void ram()
{
    if (encountcheck && walkcount >= 20)
    {
        rand = UnityEngine.Random.Range(1, 101);
        if (rand < 10 && rand > 0)
        //１０%でエンカウントする
        {
            if (charcoli.SceneFlag >= 0
            && SceneManager.GetActiveScene().name != "map")
            //エンカウントするシーン名を追加する
            {
                walkcount = 0;
                //歩数リセット
                encountflag = true;
                //エンカウント開始フラグ
                stopflag = true;
            }
        }
        encountcheck = false;
    }
}
```

図17　エンカウントシステムのスクリプト

コ　ストーリイベント

ストーリを進行するために会話文・ナレーションの表示やプレイヤの移動の制御をした。ストーリのテキストは、行数が長くなりプログラムコードが乱雑になってしまうため、スクリプトファイルとは別にテキストファイルとして管理している(図 18)。テキストファイルからプログラムで文字列を取得している(図 19)。

```
player@
ギルドマスター直々の手紙…！？@
player@
いったい何事だ…こんな最下級ランクの冒険者にいったい何の用なんだ？
player@
わからないが…呼び出されたからには行くしかないな@
player@
準備をして行こう…@
ENDEXIT@
player@
さて…行こう@
player@
確かギルドはここから西にまっすぐだったな@
ENDEXIT@
ギルド職員@
「player様ですね、こちらです@
ENDEXIT@
```

図 18　ストーリのテキストファイル

```
public void settxt()//名前入力後に実行する
{
    c = 1;
    n = 0;
    txt = ta.text;
    keeptxt = txt.Split(char.Parse("@"));
    content.text = "¥n ";
    for (i = 0; i < keeptxt.Length; i++)
    {
        keeptxt[i] = keeptxt[i].Replace("¥n", "");
        keeptxt[i] = keeptxt[i].
        Replace("player", statusdb.GetStatusLists()
    }
}
```

図 19　テキストを読み込むスクリプト

サ　Tilemap

マップ制作には「Tilemap」という Unity のコンポーネントを使用している。これは、分割された建物やフィールドのパーツを組み合わせてマップを作り上げることができる。これを使用する利点は、パーツ自体は外部のものを使用しているので、パーツのデザインを自分で用意する必要がなく、パーツを組み合わせるだけでマップを作れる点と、そのパーツを別の場所でも使える点である(図 20)。

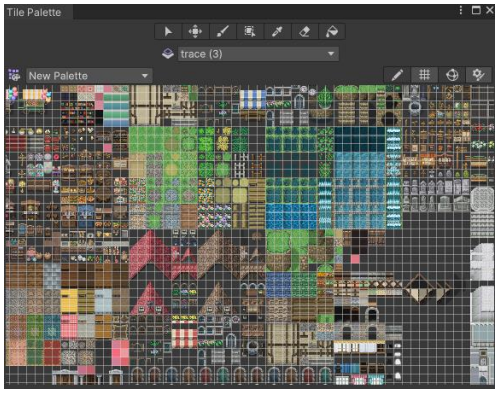

図 20　パーツの一覧

シ　SAN 値

基本的な RPG には HP、MP といった戦闘にかかわるパラメータがあるが、このゲームには他にも SAN 値という戦闘に関わるパラメータがある。SAN 値とはクトゥルフ神話 TRPG という、1981 年に制作されたテーブルトーク RPG に登場する正気度を表すパラメータであり、今回制作したゲームではその役割のほかに第二の HP としての役割もある。

SAN 値は、神話生物に出会うと減り、一度に５以上減ると確率で発狂という状態異常にかかる。５つの効果のうち１つがランダムに発動し、内容としては、お金を落とし続ける錯乱、魔法が使えなくなる忘却、敵を殴り続けてしまうヒステリ、ずっと逃げ続けてしまうパニック、手が震え目標が定まらなくなる恐怖である(図 21)。

```
switch (hakkyou)
{
        case 1://発狂して逃げ続ける
        battle.escape();
        break;
        case 2: //発狂して攻撃し続ける
        ch.atk += 10;
        battle.Click1();
        haikei.batoru();
        break;
        case 3: //発狂して魔法が打てなる
        battle.NoMagic = true;
        break;
        case 4://錯乱してお金を失う
        ch.money -= Random.Range(1, 1000);
        break;
        case 5://錯乱して標準が定まらなくなる
        battle.Ramdom = true;
        break;
}
```

図 21　発狂した場合の処理

ス　BGM・SE

(a)　BGM

ゲーム中には BGM が流れており、マップに合わせて、村やダンジョン、戦闘中などで、BGM が切り替わるようになっている。

(b)　SE

プレイヤやモンスタの攻撃、ボタンのクリック、宝箱からアイテムを手に入れるときには、それにあった SE が鳴るようになっている。また、SE を鳴らすコードが書かれている場所を明確にするために、SE は１つのスクリプトでまとめて管理している。

セ　ボート

ボートを手に入れると、水辺でボートに乗ることができ、水上を自由に移動できる。ボートにも Tilemap を使用しているので、向きを変えるときには工夫をする必要がある。ボートをそれぞれの向きに１つずつ(計４つ)用意しておき、プレイヤが向く方向によって表示・非表示を切り替えている。それにより、ボートの向きをプレイヤの向きと同期することができる。

ボートを実装するに当たって問題となったのは、水上以外の場所にもボートを出せてしまう点である。その解決策として、ボートを出せるかどうかを確認するためのオブジェクトを、ボートを出現させる先に置き、水上にしかボートを出すことができないようにした(図 22)。

```
//ボートを出す
if (collision.gameObject.name == "Water")
{
    charcoli.water = true;
    charcoli.ground = false;
}
//陸に上がる
else if (collision.gameObject.name == "Ground")
{
    charcoli.ground = true;
    charcoli.water = false;
}
```

図 22　ボートを出せるかどうかの判定

セ　無限ダンジョン

ゲームのエンドコンテンツとして無限ダンジョンという迷路型のダンジョンがある。無限ダンジョンでは迷路が自動生成されるようになっており、入るごとにランダムに生成されている。5 階層ごとに中ボス、10 階層大ボスがいる。図 23 が生成したダンジョンである。

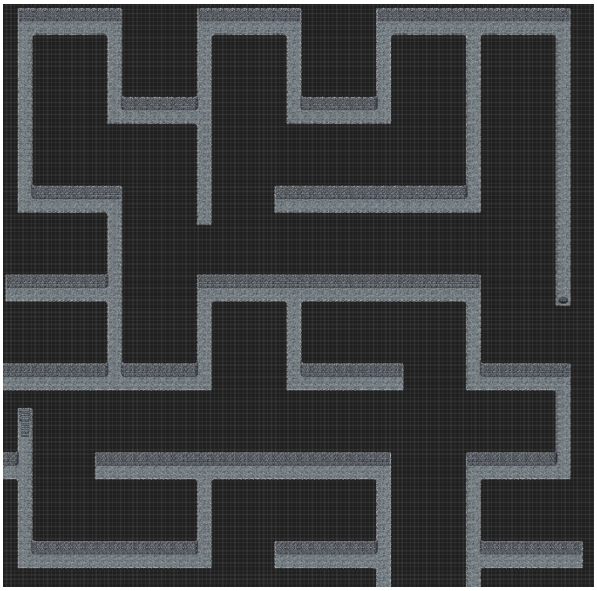

図 23　自動生成したマップ

(３)　マップ・建物

ア　マップ一覧

・村　　・家
・ショップ　　・教会
・ギルド　　・森
・廃村　　・洞窟
・無限ダンジョン

イ　マップ概要

(a)　村

村は、全体の中心となるマップであり、家やショップ、ギルド、教会など、多くの建物がある。

(b)　森

他のマップと比べて、特に広大なマップとなっている。森を通じて、廃村や洞窟、無限ダンジョンに行くことができる。また、森に入ると、敵とエンカウントするようになる。

(c)　洞窟

洞窟は、図 24 の１階層、２階層、ボス部屋の３つの階層に分かれており、全体的に水が多い地形となっている。また、ここでも敵とエンカウントするようになっており、森とは異なるモンスタが出現する。

(e)　廃村

他のマップと比べて、全体が薄暗くなっており、不穏な雰囲気を演出している。

図 24　洞窟の１階層目

（４）　その他

ゲーム制作の進捗を管理するために、GitHub や SourceTree というものを使用している。

ア　GitHub

GitHub は、「リポジトリ」というクラウドの箱にデータを保存しており、チーム内で進捗を共有するサービスである。

イ　SourceTree

SourceTree は、GitHub のデータと PC のデータをつなぐ役割をしており、他のソフトウェアと違い、コマンドを使用せず、感覚的に操作できるソフトウェアである(図 25)。

図 25　SourceTree のログ

３．研究のまとめ

安藤

今回の研究では、事前に設計をせず、必要な機能を洗い出さずにプログラミングをしてしまったため、機能を追加するとバグが発生するという事態が多々発生してしまった。

また、戦闘に関わるプログラムを分けずに１つにまとめてしまったことで、プログラムが長くなり、プログラムの可読性がとても低くなってしまい、ほかの班員を困らせてしまった。これを改善するには、機能ごとに分けることが大切だと思った。最後に、今回の課題研究で作ったゲームは機能実装に重きを置きすぎて、基盤であるゲーム自体の完成度は少し物足りないものがあった。しかし、目標はある程度達成できたと思うので、この経験は将来に活かせると思った。

北里

今回の研究では、事前に設計をしないまま制作を始めてしまった。よって、後に改修の必要があるプログラムが多く見つかり、それを修正することに時間を割かれてしまった。今後は、長期的なプロジェクトを行うときは、必ず設計をする必要があると思った。

また、データの管理がずさんであるところがあるので、はじめに、データの管理方法を決めてからプログラミングをする必要があ

ると思った。最後に、今回の研究では時間をかけた分、機能は多く実装できたと思う。ただ、ゲーム全体の完成度としては足りないところがあるので、プログラミングに取り掛かる前に構想をしっかりと練る必要があると思った。

野中

研究当初は、班員と話し合いがうまくできておらず、同じスクリプトに変更をかけてコンフリクトが頻繁に起こりデータを上書きすることがあった。それを対策するために、実装する機能ごとに役割を分け、コミュニケーションを前よりも頻繁にとるように心掛けた。その結果、上書きすることが少なくなり、同時に作業の効率も向上した。また、班員が作ったコードと機能を共有することで新しくプログラムをする必要がなくなった。

この課題研究を通して、Git での進捗管理と共同開発をするときに必要なコミュニケーションを学ぶことができた。今後、協力してものを作るときは、この経験を生かせるようにしたい。

４．参考文献

Qiita
https://qiita.com/
teratail
https://teratail.com/
侍テラコヤ
https://terakoya.sejuku.net/

・使用した素材サイト
ドット絵世界
https://yms.main.jp/page-msets/worldmap1.html
BOOTH
https://booth.pm/ja
イラストボックス
https://www.illust-box.jp/
フリーBGM DOVA-SYNDROME
https://dova-s.jp/
効果音ラボ
https://soundeffect-lab.info/
On-Jin～音人～
https://on-jin.com/about.php
Material Forward
https://materialforwardvfx.wixsite.com/materialforward
Pixiv
https://www.pixiv.net/
illustAC
https://www.ac-illust.com/
DOT ILLUST
https://dot-illust.net/