```
//実行条件：プロジェクト/プロパティ/詳細/マルチバイト文字セットの使用
#pragma warning(disable:4996)//vs2019がエラーを報告しないようにし、プログラムをスムーズに実行できるようにする
#pragma comment(lib,"Winmm.lib")
#include <stdio.h>
#include <graphics.h>
#include <math.h>
#include <conio.h>
#include <stdlib.h>
#include "EasyXPng.h"
#include <mmsystem.h>
#define width 840
#define height 700//ゲームの画面サイズ
#define towernum 9//防御塔の数

int gif = 0;
TCHAR enemy[10];//テキスト出力
IMAGE mainmenu/*メインメニュー*/, door_l, door_r, pause, victory, lose;
int enemynum = 0;
int musicon = 1;

//関数を宣言する
void Restart(), Quit(), Door(), Pause(), Victory(), Lose(), Playmusic();
void BiJin(float* x1, float* y1, float* x2, float* y2, float* speed)
{
    float speedx = *speed, speedy = *speed, dis, disy = *y2 - *y1, disx = *x2 - *x1;
    if (disx != 0 && disy != 0)
    {
        dis = sqrt(disy * disy + disx * disx);
        speedy = fabs(disy * *speed / dis);
        speedx = fabs(disx * *speed / dis);
    }
    if (speedx == 0)
        speedx = *speed;
    if (speedy == 0)
        speedy = *speed;
    if (disx > 0)
        *x1 += speedx;
    else
        *x1 -= speedx;
    if (disy > 0)
        *y1 += speedy;
    else
        *y1 -= speedy;
}
//攻撃アニメーション
class WEAPON
{
public:
    IMAGE image;
    float x, y;
    char imagename[20];
    int num, hp = 1, flyingtime;
    void Draw()
    {
        sprintf(imagename, "weapon_%d.png", num);
```

```
        if (num == 2)
        {
            sprintf(imagename, "weapon_%d_%d.png", num, gif / 100 % 2);
            if (hp <= 0)
                sprintf(imagename, "weapon_%d_2.png", num);
        }
        loadimage(&image, imagename);
        putimagePng(x, y, &image);
    }
};
//防御塔関連
class TOWER
{
public:
    IMAGE image;//関連画像
    int value, price, num, lev, attacking, time;//関連データ
    float ATK, radius, ATKS;//防御塔の属性
    float x, y;//座標
    WEAPON weapon;
    // タワー付録-タワーメニュー関連
    IMAGE menucircle, towerUI[2], guide;
    int menuopen, guideshow[8];//判断変数
    char imagename[20], guidename[20], filename[20];//ファイル名
    void Update()//防御塔の属性を更新するために、防御塔が変更されたときに呼び出される
    {
        sprintf(filename, "towerdata_%d_%d.txt", num, lev);
        FILE* ftower = fopen(filename, "r");
        fscanf(ftower, "%d %d %f %f %f ", &price, &value, &ATK, &radius, &ATKS);
        fclose(ftower);
        ATK += rand() % 4;
        if (num == 2)
            weapon.hp = lev * 50;
    }
    void Draw()//防御塔を描き
    {
        if (lev != 0)
            sprintf(imagename, "tower_%d_%d.png", num, lev);//フォーマット変換
        else
            sprintf(imagename, "notower.png");
        loadimage(&image, imagename);//画像をインポートする
        putimagePng(x - 36, y - 21, &image);//防御塔の画像
    }
    void Menu()//メニューを描き
    {
        if (menuopen == 1)//防御塔メニューを開くかどうかを決定する
        {
            putimagePng(x - 50, y - 50, &menucircle);
            if (lev == 0)
            {
                for (int i = 1; i <= 4; i++)
                {//マスクマップの描画
                    putimage(x - 22 + 38 * pow(-1, i), y - 22 + 38 * pow(-1, (i + 1) / 2),
                      46, 41, &towerUI[0], 46 * (i - 1), 0, SRCAND);
                    putimage(x - 22 + 38 * pow(-1, i), y - 22 + 38 * pow(-1, (i + 1) / 2),
                      46, 41, &towerUI[1], 46 * (i - 1), 0, SRCPAINT);
                }
```

```
            }
            else if (lev < 3)
            {
                putimage(x - 23, y - 68, 46, 41, &towerUI[0], 46 * 4, 0, SRCAND);
                putimage(x - 23, y - 68, 46, 41, &towerUI[1], 46 * 4, 0, SRCPAINT);
                putimage(x - 21, y + 27, 46, 41, &towerUI[0], 46 * 6, 0, SRCAND);
                putimage(x - 21, y + 27, 46, 41, &towerUI[1], 46 * 6, 0, SRCPAINT);
            }
            else
            {
                putimage(x - 23, y - 68, 46, 41, &towerUI[0], 46 * 5, 0, SRCAND);
                putimage(x - 23, y - 68, 46, 41, &towerUI[1], 46 * 5, 0, SRCPAINT);
                putimage(x - 21, y + 27, 46, 41, &towerUI[0], 46 * 6, 0, SRCAND);
                putimage(x - 21, y + 27, 46, 41, &towerUI[1], 46 * 6, 0, SRCPAINT);
            }
            //防御塔UIを表示する
            for (int i = 1; i <= 4; i++)
                if (guideshow[i] == 1)
                {
                    if (x + 40 * pow(-1, i) >= 610)
                        putimagePng(x + 15 + 38 * pow(-1, i) - 257, y + 10 + 38 * pow(-1, (i + 1) / 2), &guide);
                    else if (y >= 484)
                        putimagePng(x + 15 + 38 * pow(-1, i), y + 10 + 38 * pow(-1, (i + 1) / 2) - 190, &guide);
                    else
                        putimagePng(x + 15 + 38 * pow(-1, i), y + 10 + 38 * pow(-1, (i + 1) / 2), &guide);
                }
            if (guideshow[5] == 1)
                if (x <= 400)
                    putimagePng(x + 13, y - 34, &guide);
                else
                    putimagePng(x - 244, y - 34, &guide);
        }
    }
};
//モンスター関連
class ENEMY
{
public:
    IMAGE image, hp;
    int num, attacted, value, aim = 1;//関連データ
    float x, y;//モンスター座標
    float HP, Hp, ATK, DEF, speed;//モンスター属性
    float w, h;//画像の長さと幅
    char imagename[20];//画像名
    void Update()
    {
        char filename[20];
        sprintf(filename, "enemydata_%d.txt", num);
        FILE* fenemy = fopen(filename, "r");
        fscanf(fenemy, "%d %f %f %f %f %f", &value, &Hp, &HP, &ATK, &DEF, &speed);
        fclose(fenemy);
    }
    void Draw()
```

```
    {
        loadimage(&hp, "HP.png", Hp, 3);
        sprintf(imagename, "enemy_%d_%d.png", num, gif / 100 % 2);
        loadimage(&image, imagename);
        w = image.getwidth();
        h = image.getheight();
        putimagePng(x - w / 2, y - h, &image);
        putimage(x - Hp / 4, y - h - 10, Hp / 2, 3, &hp, (Hp - HP) / 2, 0);
    }
};
//バトルステージ
class MAP
{
public:
    IMAGE image;//バックグラウンド
    int lev, money, heart, enemywave, kill;
    float spot_x[3][10], spot_y[3][10];//キーポイント、モンスターの移動ルートをマップする
    char ht[2], my[4], ew[10];//関連するテキスト出力
    TOWER tower[towernum];//防御塔の数
    ENEMY enemy[99];//モンスター
    char imagename[20], mapname[20], spotname[20];//ファイル名
    void Start()
    {
        sprintf(imagename, "level_%d.png", lev);
        sprintf(mapname, "level_%d.txt", lev);//フォーマット変換を使用し、バッチインポート
          を実現する
        loadimage(&image, imagename);
        money = 320;
        heart = 20;
        kill = 0;
        enemynum = 0;//データリセット
        FILE* fmap = fopen(mapname, "r");//ファイルからタレット座標を読み取る
        for (int i = 0; i < 3; i++)
        {
            sprintf(spotname, "aim_%d_%d.txt", lev, i % 3);
            FILE* faim = fopen(spotname, "r");
            for (int j = 0; j < 7; j++)
            {
                fscanf(faim, "%f %f ", &spot_x[i][j], &spot_y[i][j]);
            }
            fclose(faim);
        }
        for (int i = 0; i < towernum; i++)
        {
            loadimage(&tower[i].menucircle, "menucircle.png", 100, 100);
            loadimage(&tower[i].towerUI[0], "towerUIy.png", 322, 41);
            loadimage(&tower[i].towerUI[1], "towerUI.png", 322, 41);
            tower[i].lev = 0;
            fscanf(fmap, "%f %f ", &tower[i].x, &tower[i].y);//座標を読む
            tower[i].weapon.x = tower[i].x;
            tower[i].weapon.y = tower[i].y;
        }
        for (int i = 0; i < 99; i++)
        {
            enemy[i].aim = 1;
            enemy[i].x = spot_x[0][1];
```

```
            enemy[i].y = spot_y[0][1];
        }
        fclose(fmap);//ファイルを閉じる
        BeginBatchDraw();
    }   //マップ描画機能
    void Draw()//マップを描き//メインドロー関数
    {
        cleardevice();//クリアスクリーン
        putimage(0, 0, &image);//バックグラウンド
        for (int i = 0; i < towernum; i++)
            tower[i].Draw();//防御塔を描き
        for (int i = 0; i < towernum; i++)
            if (tower[i].attacking == 1 && tower[i].lev > 0)
                tower[i].weapon.Draw();
        for (int i = 0; i < enemynum; i++)
            if (enemy[i].HP > 0)
                enemy[i].Draw();//モンスターを描き

        for (int i = 0; i < towernum; i++)
            tower[i].Menu();//防御塔メニューを描き
    }
    void Update()
    {
        if (gif % (1000 - enemynum * 10) == 0 && enemynum <= 99)
        {
            enemynum++;
            enemy[enemynum].num = 1;
            if (enemynum % 10 == 0)
                enemy[enemynum].num = 2;
            if (enemynum % 30 == 0)
                enemy[enemynum].num = 3;
            enemy[enemynum].Update();//新しく入ったモンスターとそれらを更新する
        }
        for (int i = 0; i < enemynum; i++)
        {
            if (enemy[i].HP > 0)//モンスターが生きている場合のみ
            {
                BiJin(&enemy[i].x, &enemy[i].y, &spot_x[i % 3][enemy[i].aim], &spot_y[i %
                  3][enemy[i].aim], &enemy[i].speed);
                if (fabs(enemy[i].x - spot_x[i % 3][enemy[i].aim]) < 1 && fabs(enemy[i].y -
                   spot_y[i % 3][enemy[i].aim]) <= 1)
                    enemy[i].aim++;
                if (fabs(enemy[i].x - spot_x[i % 3][6]) < 1 && fabs(enemy[i].y - spot_y[i %
                   3][6]) < 1)
                    heart--;
            }
        }
        for (int i = 0; i < towernum; i++)
        {
            for (int j = 0; j < enemynum; j++)
            {
                float x = enemy[j].x - enemy[j].w / 2, y = enemy[j].y - enemy[j].h;
                if (tower[i].lev == 0)
                    tower[i].attacking = 0;
                if (tower[i].lev > 0 && enemy[j].HP > 0)
                {
```

```
                    if (sqrt((tower[i].x - x) * (tower[i].x - x) + (tower[i].y - y) *
                      (tower[i].y - y)) < tower[i].radius)
                    {//モンスターが攻撃範囲に入ったとき
                        tower[i].attacking = 1;
                        enemy[j].attacted = 1;
                    }
                    if (tower[i].attacking == 1 && enemy[j].attacted == 1 && tower
                      [i].weapon.hp > 0
                        && sqrt((tower[i].x - x) * (tower[i].x - x) + (tower[i].y - y) *
                       (tower[i].y - y)) < tower[i].radius)//防御塔が攻撃していて、モンス
                       ターが攻撃を受けているとき
                        BiJin(&tower[i].weapon.x, &tower[i].weapon.y, &x, &y, &tower
                       [i].ATKS);
                    if (tower[i].num == 2)
                    {
                        if (fabs(tower[i].weapon.x - x) <= enemy[j].w / 2 && fabs(tower
                       [i].weapon.y - y) <= enemy[j].h / 2 && tower[i].weapon.hp > 0)
                        {
                            if (gif % 200 == 0)
                                tower[i].weapon.hp -= enemy[j].ATK;
                            if (gif % 300 == 0)
                                enemy[j].HP -= tower[i].ATK + rand() % 3;
                        }
                        if (tower[i].num == 2 && tower[i].weapon.hp <= 0)
                        {
                            tower[i].time++;
                            if (tower[i].time == 20000)
                                tower[i].weapon.hp = 50 * tower[i].lev;
                        }
                    }
                    if (tower[i].num != 2)
                    {
                        if (sqrt((tower[i].x - x) * (tower[i].x - x) + (tower[i].y - y) *
                       (tower[i].y - y)) > tower[i].radius)
                        {
                            enemy[j].attacted = 0;
                            enemy[j + 1].attacted = 1;
                        }
                        if (fabs(tower[i].weapon.x - x) <= enemy[j].w / 2 && fabs(tower
                       [i].weapon.y - y) <= enemy[j].h / 2)
                        {//防御塔の攻撃がモンスターと効果的に衝突したとき
                            tower[i].weapon.x = tower[i].x - 5;
                            tower[i].weapon.y = tower[i].y - 16;
                            enemy[j].HP -= tower[i].ATK - enemy[j].DEF;
                        }
                        if (enemy[j].HP <= 0)
                        {
                            kill++;
                            money += enemy[j].value + rand() % 4;
                            enemy[j].attacted = 0;
                        }
                    }
                }
            }
        }
    }
```

```
} map;
//リソースの初期化とインポート
void Start()
{
    //グローバルリソースのインポート
    loadimage(&pause, "pause.png", 428, 312);
    loadimage(&door_l, "doorl.png", width / 2, height);
    loadimage(&door_r, "doorr.png", width / 2, height);
    loadimage(&mainmenu, "mainmenu.png");
    loadimage(&victory, "victory.png", 384, 384);
    loadimage(&lose, "lose.png", 384, 384);
    initgraph(width, height);
    Playmusic();
    setbkmode(TRANSPARENT);//すべてのテキストの透過的な出力
    BeginBatchDraw();//バッチ描画を開始する
}
//入力に関係のないものの更新
void UpdateWithoutInput()
{
    map.Update();
    map.Draw();
    settextstyle(40, 0, _T("Impact"));
    sprintf(map.ht, _T("%d"), map.heart);
    outtextxy(40, 0, map.ht);
    sprintf(map.my, _T("%d"), map.money);
    outtextxy(160, 0, map.my);
    sprintf(map.ew, _T("WAVE %d/16"), enemynum / 6);
    outtextxy(60, 40, map.ew);
    if (map.kill == 99 && map.heart > 0)
        Victory();
    if (map.heart <= 0)
        Lose();
}
//入力に関連するものの更新
void UpdateWithInput()
{//マウス操作と対応する判定変数の変更
    if (MouseHit())
    {
        MOUSEMSG m = GetMouseMsg();//マウス情報を取得する
        for (int i = 0; i < towernum; i++)
        {
            if (map.tower[i].menuopen == 0)//防御塔メニューが開いていないとき
            {
                if (fabs(m.x - map.tower[i].x) <= 36 && fabs(m.y - map.tower[i].y) <= 21
                    && fabs(m.x - 0) >= 36 && fabs(m.y - 0) >= 21 && m.uMsg ==
                      WM_LBUTTONDOWN)
                {
                    //他のメニューを閉じ、このメニューのみを開き
                    for (int j = 0; j < towernum; j++)
                    {
                        if (j == i)
                            map.tower[j].menuopen = 1;
                        else
                            map.tower[j].menuopen = 0;
                    }
                }
```

```
            }
            else
            {
                if (map.tower[i].lev == 0)
                {
                    for (int j = 1; j <= 4; j++)
                        if (fabs(m.x - map.tower[i].x - 40 * pow(-1, j)) < 20 && fabs(m.y -
                        map.tower[i].y - 40 * pow(-1, (j + 1) / 2)) < 20)
                        {
                            if (map.tower[i].guideshow[j] == 1 && m.uMsg == WM_LBUTTONDOWN)
                            {
                                map.tower[i].weapon.num = map.tower[i].num = j;//防御塔番号
                            を確定する
                                map.tower[i].lev++;//偽のアップグレードは価格を取得する
                                map.tower[i].Update();
                                if (map.money >= map.tower[i].price)
                                    map.money -= map.tower[i].price;//お金があれば差し引く
                                else
                                    map.tower[i].lev--;//お金がなければレベルを戻す
                            }
                            else
                            {
                                sprintf(map.tower[i].guidename, "guide_%d_1.png", j);
                                loadimage(&map.tower[i].guide, map.tower[i].guidename, 230,
                            166);
                                for (int k = 1; k <= 4; k++)
                                    if (k == j)
                                        map.tower[i].guideshow[k] = 1;
                                    else
                                        map.tower[i].guideshow[k] = 0;
                            }
                        }
                        else
                            map.tower[i].guideshow[j] = 0;
                }
                else
                {
                    if (fabs(m.x - map.tower[i].x) < 20 && fabs(m.y - map.tower[i].y + 50)
                      < 20 && map.tower[i].lev < 3)
                    {
                        if (map.tower[i].guideshow[5] == 1 && m.uMsg == WM_LBUTTONDOWN)
                        {
                            map.tower[i].lev++;
                            map.tower[i].Update();
                            if (map.money >= map.tower[i].price)
                                map.money -= map.tower[i].price;
                            else
                            {
                                map.tower[i].lev--;
                                map.tower[i].Update();
                            }
                        }
                        else
                        {//マウスを合わせると、テキストの紹介が表示される
                            sprintf(map.tower[i].guidename, "guide_%d_%d.png", map.tower
                        [i].num, map.tower[i].lev + 1);
```

```
                        loadimage(&map.tower[i].guide, map.tower[i].guidename, 230,
                    166);
                        map.tower[i].guideshow[5] = 1;
                    }
                }
                else if (fabs(m.x - map.tower[i].x) < 13 && fabs(m.y - map.tower[i].y -
                    50) < 13 && m.uMsg == WM_LBUTTONDOWN)
                {//解体防御塔
                    map.money += map.tower[i].value;
                    map.tower[i].lev = 0;
                }
                else
                    map.tower[i].guideshow[5] = 0;
            }
            if (m.uMsg == WM_LBUTTONDOWN)
            {
                map.tower[i].menuopen = 0;
                for (int j = 0; j < 8; j++)
                    map.tower[i].guideshow[j] = 0;
            }
        }
    }
    if (m.uMsg == WM_LBUTTONDOWN && fabs(m.x - 786) <= 14 && fabs(m.y - 53) <= 13)
        Pause();
  }
}
//音楽を再生
void Playmusic()
{
    mciSendString("open bkmusic.mp3 alias bkmusic", NULL, 0, NULL);
    mciSendString("play bkmusic", NULL, 0, NULL);
}
//インターフェイス切り替えアニメーション
void Door()
{
    for (int i = width; i >= width / 2; i -= 2)
    {
        cleardevice();
        putimage(i, 0, &door_r);
        putimage(width / 2 - i, 0, &door_l);
        FlushBatchDraw();
    }
    Sleep(800);
    for (int i = width / 2; i <= width; i += 2)
    {
        cleardevice();
        putimage(i, 0, &door_r);
        putimage(width / 2 - i, 0, &door_l);
        FlushBatchDraw();
    }
}
//インターフェイスを一時停止
void Pause()
{
    while (1)
    {
```

```
        putimagePng(206, 194, &pause);
        if (MouseHit())
        {
            MOUSEMSG m = GetMouseMsg();
            if (m.uMsg == WM_LBUTTONDOWN && fabs(m.y - 263) <= 15)
            {
                if (fabs(m.x - 263) <= 18 && musicon == 1)
                {
                    mciSendString("pause bkmusic", NULL, 0, NULL);
                    musicon = 0;
                }
                else if (fabs(m.x - 263) <= 18 && musicon == 0)
                {
                    mciSendString("resume bkmusic", NULL, 0, NULL);
                    musicon = 1;
                }
            }
            if (m.uMsg == WM_LBUTTONDOWN && fabs(m.x - 420) <= 110)
            {
                if (fabs(m.y - 263) <= 30)
                    Restart();
                else if (fabs(m.y - 340) <= 30)
                    break;
                else if (fabs(m.y - 420) <= 30)
                {
                    Door();
                    Quit();
                }
            }
        }
        FlushBatchDraw();
    }
}
//勝利インターフェース
void Victory()
{
    while (1)
    {
        putimagePng(228, 158, &victory);
        if (MouseHit())
        {
            MOUSEMSG m = GetMouseMsg();
            if (m.uMsg == WM_LBUTTONDOWN)
            {
                if (fabs(m.x - 421) <= 183 && fabs(m.y - 395) <= 25)
                {
                    if (map.lev == 9)
                        Quit();
                    map.lev++;
                    Restart();
                }
                else if (fabs(m.x - 420) <= 65 && fabs(m.y - 466) <= 22)
                    Restart();
            }
        }
        FlushBatchDraw();
```

```
    }
}
//失敗インターフェース
void Lose()
{
    while (1)
    {
        putimagePng(228, 158, &lose);
        if (MouseHit())
        {
            MOUSEMSG m = GetMouseMsg();
            if (m.uMsg == WM_LBUTTONDOWN)
            {
                if (fabs(m.x - 421) <= 183 && fabs(m.y - 395) <= 25)
                {
                    if (map.lev == 9)
                        Quit();
                    Restart();
                }
                else if (fabs(m.x - 420) <= 65 && fabs(m.y - 466) <= 22)
                    Quit();
            }
        }
        FlushBatchDraw();
    }
}
//メインメニュー
void Mainmenu()
{
    while (1)//ループから飛び出してバトルステージを変える
    {
        putimage(0, 0, &mainmenu);//バトルステージインターフェース
        FlushBatchDraw();
        if (MouseHit())
        {
            MOUSEMSG m = GetMouseMsg();//マウス情報を取得する
            if (m.uMsg == WM_LBUTTONDOWN)//左ボタンを押したとき
            {
                if (fabs(m.x - 270) <= 10 && fabs(m.y - 170) <= 20)
                    map.lev = 1;
                else if (fabs(m.x - 330) <= 10 && fabs(m.y - 220) <= 20)
                    map.lev = 2;
                else if (fabs(m.x - 270) <= 10 && fabs(m.y - 310) <= 20)
                    map.lev = 3;
                else if (fabs(m.x - 150) <= 10 && fabs(m.y - 380) <= 20)
                    map.lev = 4;
                else if (fabs(m.x - 300) <= 10 && fabs(m.y - 510) <= 20)
                    map.lev = 5;
                else if (fabs(m.x - 400) <= 10 && fabs(m.y - 420) <= 20)
                    map.lev = 6;
                else if (fabs(m.x - 590) <= 10 && fabs(m.y - 435) <= 20)
                    map.lev = 7;
                else if (fabs(m.x - 575) <= 10 && fabs(m.y - 310) <= 20)
                    map.lev = 8;
                else if (fabs(m.x - 720) <= 10 && fabs(m.y - 295) <= 20)
                    map.lev = 9;
```

```
                else
                    continue;
                Sleep(1000);//短い遅延
                break;//ループから飛び出してバトルに入る
            }
        }
    }
}
//ゲーム中//バトルを再開するために使用されます
void Restart()
{
    Door();
    map.Start();//各バトルでリソースを初期化してロードします
    map.Draw();
    gif = 0;
    while (1)
    {
        UpdateWithInput();//入力に関連するものの更新
        UpdateWithoutInput();//入力に関係のないものの更新
        FlushBatchDraw();
        gif++;
    }
}
//ゲーム本体
void Quit()
{
    while (1)
    {
        Mainmenu();//メインメニューインターフェース
        Restart();
    }
}
//メイン関数
int main()
{
    Start();//リソースの初期化とロード
    Quit();
    system("pause");
    closegraph;//キャンバスを閉じる
    return 0;
}
```