6月3日の授業で作成したところまでのサンプルソースです。
作成が追いついていない方、どうしてもエラーが発生して困っている方は
このソースを参考にして、次回の授業日の6月8日までに動作する様にしておいてください。
※チャプター２でのサンプルソースの配布は今回が最後となります。

```
//-------------------------------------------------------------------------------
// シューティングゲーム
// 氏　名：谷口　知士郎
//-------------------------------------------------------------------------------
#include <DxLib.h>

//-----------------
// 定数定義
//-----------------
const int WINDOW_SIZE_WID = 800;            // ウィンドウの横サイズ
const int WINDOW_SIZE_HIG = 600;            // ウィンドウの縦サイズ

const int PLAYER_SIZE_WID = 64;             // プレイヤー画像の横サイズ
const int PLAYER_SIZE_HIG = 64;             // プレイヤー画像の縦サイズ
const int PLAYER_MOVE_SPEED_X = 4;          // プレイヤー機体のX方向の移動量
const int PLAYER_MOVE_SPEED_Y = 4;          // プレイヤー機体のY方向の移動量

const int PLAYER_BULLET_WID = 32;           // 敵画像の横サイズ
const int PLAYER_BULLET_HIG = 32;           // 敵画像の縦サイズ
const int PLAYER_BULLET_MOVE = 8;           // プレイヤー弾の移動速度

const int ENEMY_SIZE_WID = 96;              //敵画像の横サイズ
const int ENEMY_SIZE_HIG = 96;              //敵画像の縦サイズ
const int ENEMY_MOVE_SPEED = 2;             //敵の移動量

//-----------------
// 変数定義
//-----------------
int playerImage;                // 自機の画像のハンドル番号
int playerPosX;                 // 自機のX座標
int playerPosY;                 // 自機のY座標
bool playerFlg;                 // プレイヤーの表示状態管理するフラグ(true:表示、false:非表示)

int enemyImage;                 // 敵キャラの画像のハンドル番号
int enemyPosX;                  // 敵のX座標
int enemyPosY;                  // 敵のY座標
bool enemyFlg;                  // 敵の表示状態管理するフラグ(true:表示、false:非表示)

int playerBulletImage;          // プレイヤー弾の画像のハンドル番号
int playerBulletPosX;           // プレイヤー弾のX座標
int playerBulletPosY;           // プレイヤー弾のY座標
bool shotFlg;                   // 弾の発射状態を管理する(true=発射中, false=未発射)

int enemyClrCounter;

bool bulletHitFlg = false;
```

```
//-----------------
// WinMain関数
//-----------------
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
  //-----------------
  // システム設定
  //-----------------
  SetWindowText("000000　谷口知士郎");                    // ウィンドウのタイトルの設定
  SetGraphMode(WINDOW_SIZE_WID, WINDOW_SIZE_HIG, 32);     // ウィンドウのサイズと色モードの設定
  ChangeWindowMode(true);
                  // ウィンドウの表示モード(true=ウィンドウモード/false=フルスクリーン)
  int err = DxLib_Init();       // DXライブラリの初期化処理
  if (err == -1) {
    return -1;                  // DXライブラリの初期化に失敗したので、システムをエラー終了する
  }

  //---------------------------
  // グラフィックの登録
  //---------------------------
  // 自機画像の読込み
  playerImage = LoadGraph("image/player.png");
  if (playerImage == -1) {
    return -1;                  // 画像読み込み失敗なので、エラー終了
  }

  // 敵画像の読込み
  enemyImage = LoadGraph("image/enemy.png");
  if (enemyImage == -1) {
    return -1;                  // 画像読み込みに失敗したので、エラー終了
  }

  // 弾画像の読み込み
  playerBulletImage = LoadGraph("image/bullet.png");
  if (playerBulletImage == -1) {
    return -1;                  // 画像読み込みに失敗したので、エラー終了
  }

  //-----------------
  // 変数の初期化
  //-----------------
  // プレイヤーの初期位置
  playerPosX = (WINDOW_SIZE_WID / 2) - (PLAYER_SIZE_WID / 2);
  playerPosY = (WINDOW_SIZE_HIG / 2) - (PLAYER_SIZE_HIG / 2);
  playerFlg = true;

  // 敵の初期位置
  enemyPosX = (WINDOW_SIZE_WID / 2) - (ENEMY_SIZE_WID / 2);
  enemyPosY = 0;
  enemyFlg = true;

  // プレイヤーの弾
  playerBulletPosX = playerBulletPosY = 0;
  shotFlg = false;

  enemyClrCounter = 0;
```

```
//-----------------
// ゲームループ
//-----------------
while (ProcessMessage() == 0 && CheckHitKey(KEY_INPUT_ESCAPE) == 0) {
   if (playerFlg == true) {
      //--------------------------
      // プレイヤーの移動処理
      //--------------------------
      // 右キーが押されたか
      if (CheckHitKey(KEY_INPUT_RIGHT) == 1) {
         playerPosX += PLAYER_MOVE_SPEED_X;
      }
      // 左キーが押されたか
      if (CheckHitKey(KEY_INPUT_LEFT) == 1) {
         playerPosX -= PLAYER_MOVE_SPEED_X;
      }
      // 上キーが押されたか
      if (CheckHitKey(KEY_INPUT_UP) == 1) {
         playerPosY -= PLAYER_MOVE_SPEED_Y;
      }
      // 下キーが押されたか
      if (CheckHitKey(KEY_INPUT_DOWN) == 1) {
         playerPosY += PLAYER_MOVE_SPEED_Y;
      }

      // 自機の現在位置がウィンドウ外に飛び出していないかのチェックを行う。
      // 左端のチェック
      if (playerPosX < 0) {
         playerPosX = 0;
      }
      // 右端のチェック
      if (playerPosX > (WINDOW_SIZE_WID - PLAYER_SIZE_WID)) {
         playerPosX = (WINDOW_SIZE_WID - PLAYER_SIZE_WID);
      }
      // 上端のチェック
      if (playerPosY < 0) {
         playerPosY = 0;
      }
      // 下端のチェック
      if (playerPosY > (WINDOW_SIZE_HIG - PLAYER_SIZE_HIG)) {
         playerPosY = (WINDOW_SIZE_HIG - PLAYER_SIZE_HIG);
      }

      //--------------------------
      // プレイヤー弾の処理
      //--------------------------
      if (shotFlg == false) {
         // プレイヤー弾が未発射状態なので、弾を撃つ事ができる
         if (CheckHitKey(KEY_INPUT_SPACE) == 1) {
            playerBulletPosX = playerPosX + (PLAYER_SIZE_WID - PLAYER_BULLET_WID) / 2;
            playerBulletPosY = playerPosY - PLAYER_BULLET_HIG;
            shotFlg = true;
         }
      }
   } else {
      // プレイヤーが非表示状態なので、再表示
      playerPosX = (WINDOW_SIZE_WID / 2) - (PLAYER_SIZE_WID / 2);
      playerPosY = WINDOW_SIZE_HIG - PLAYER_SIZE_HIG;
      playerFlg = true;
   }
```

```
//--------------------------
// プレイヤー弾の移動処理
//--------------------------
if (shotFlg == true) {
  // プレイヤー弾が発射されている状態
  // プレイヤー弾を移動させる
  playerBulletPosY -= PLAYER_BULLET_MOVE;

  if (playerBulletPosY < 0) {
    // プレイヤー弾がウィンドウ外に出たので、未発射状態にする
    shotFlg = false;
  }
}

//--------------------------
// 敵の処理
//--------------------------
if (enemyFlg) {
  // 敵の移動処理
  enemyPosY += ENEMY_MOVE_SPEED;
  if (enemyPosY >= WINDOW_SIZE_HIG) {
    // 敵がウィンドウの外に飛び出したので消す
    enemyFlg = false;
    enemyClrCounter = 120;
  }
} else {
  // 敵が非表示になっているので、再度画面上部に表示する
  bulletHitFlg = false;
  enemyClrCounter++;
  if (enemyClrCounter > 120) {
    enemyPosX = WINDOW_SIZE_WID / 2 - ENEMY_SIZE_WID / 2;
    enemyPosY = 0;
    enemyFlg = true;

    enemyClrCounter = 0;
  }
}

//--------------------------
//当たり判定
//--------------------------
// 弾と敵
if (enemyFlg == true && shotFlg == true) {
  // 敵と弾の両方ともが表示されている
  if ((enemyPosY + ENEMY_SIZE_HIG > playerBulletPosY)        // ①敵の下 ＞ 弾の上
    && (enemyPosY < playerBulletPosY + PLAYER_BULLET_HIG)    // ②敵の上 ＜ 弾の下
    && (enemyPosX < playerBulletPosX + PLAYER_BULLET_WID)    // ③敵の左 ＜ 弾の右
    && (enemyPosX + ENEMY_SIZE_WID > playerBulletPosX)       // ④敵の右 ＞ 弾の左
    ) {
    enemyFlg = false;                // 敵を倒す
    shotFlg = false;                 // 弾を消す
    enemyClrCounter = 0;
    bulletHitFlg = true;
  } else {
    bulletHitFlg = false;
  }
}
```

```
    // 敵とプレイヤー
    if (playerFlg == true && enemyFlg == true) {
      // 敵とプレイヤーが両方表示されている
      if ((enemyPosY + ENEMY_SIZE_HIG > playerPosY)              // ①敵の下 ＞ プレイヤーの上
        && (enemyPosY < playerPosY + PLAYER_SIZE_HIG)            // ②敵の上 ＜ プレイヤーの下
        && (enemyPosX < playerPosX + PLAYER_SIZE_WID)            // ③敵の左 ＜ プレイヤーの右
        && (enemyPosX + ENEMY_SIZE_WID > playerPosX)             // ④敵の右 ＞ プレイヤーの左
        ) {
        playerFlg = false;              // プレイヤーを消す

        bulletHitFlg = true;
      }
      else {
        bulletHitFlg = false;
      }

    }

    //-------------------------
    // 描画処理
    //-------------------------
    SetDrawScreen(DX_SCREEN_BACK);      // 描画する画面を裏の画面に設定する
    ClearDrawScreen();                  // 描画する画面の内容を消去する


    // 対角線を引く
    DrawLine(0, 0, WINDOW_SIZE_WID, WINDOW_SIZE_HIG, 0xffffff);
    DrawLine(0, WINDOW_SIZE_HIG, WINDOW_SIZE_WID, -1, 0xffffff);


    // プレイヤーの表示
    if (playerFlg == true) {
      DrawGraph(playerPosX, playerPosY, playerImage, true);
    }

    // 敵の表示
    //-------------------------
    if (enemyFlg == true) {
      DrawGraph(enemyPosX, enemyPosY, enemyImage, true);
    }

    // プレイヤー弾の表示
    //-------------------------
    if (shotFlg) {
      DrawGraph(playerBulletPosX, playerBulletPosY, playerBulletImage, true);
    }


    ScreenFlip();     // 裏の画面を表の画面に瞬間コピー
  }

  //---------------------
  // システムの終了処理
  //---------------------
  DxLib_End();        // DXライブラリの終了処理
  return 0;           // ゲームの正常終了
}
```