# **Chapter 1 デカルト座標系**

## **1.1 2Dデカルト座標**

　デカルトとは直角を表す言葉です。皆さんが中学校の数学などで見たことがあるであろう、下のような図1-1が2Dデカルト座標系です。

図1-1



## **1.1.2 原点**

　すべての2Dデカルト座標系には、原点と呼ばれる座標系の中心を表す特別な場所を持っています。

## **1.1.3　軸**

　すべての2Dデカルト座標系には、原点を通る２つの直線を持っています。それぞれの線は軸として知られています。その軸は2Dデカルト座標系であれば、x軸、y軸と呼ばれることが多いです。

## **1.1.4 デカルト座標系を用いて、場所を指定する**

　デカルト座標系を使えば、2Dゲームのオブジェクトの表示されている場所を、数字で指定することができます。下記の図1-2を見てみてください。

　図1-2



ヨッシーー

マリオ

この図であればマリオの場所は、X軸上に2.5、Y軸上に0.0で表すことができます。ヨッシーの場所は、X軸上に1.0、Y軸上に2.0あたりでしょうか。

## **1.1.5 軸の向き**

　我々はxの＋は右方向、yの+は上方向と習慣的に覚えているかもしれません。しかし、xの+を左方向、yの+を下方向とすることもできます。例えば、ウィンドウプログラムでは図1-3のように、yの+が下方向になっていることがあります。

図1-3



ここで重要なのは、ウィンドウプログラムだとYの方向が違うということを覚えることではありません。次のことをしっかりと頭に入れておいてください。

**「右方向がXの＋、上方向がYの+であるとは限らない！」**

ただし、この後の2Dデカルト座標系での話は、みなさんが分かりやすいように、右が+x、上が+yとして話を進めていきます。

## **1.1.6 プログラムでのオブジェクトの座標の指定の仕方**

　キャラクターを画面に表示するためには、**「どこに表示するのか？」**ということをコンピュータに教えてやる必要があります。多くのゲームでは、この場所に指定にベクトル構造体(もしくはベクトルクラス)を使ってデカルト座標系での位置をコンピュータに教えます。

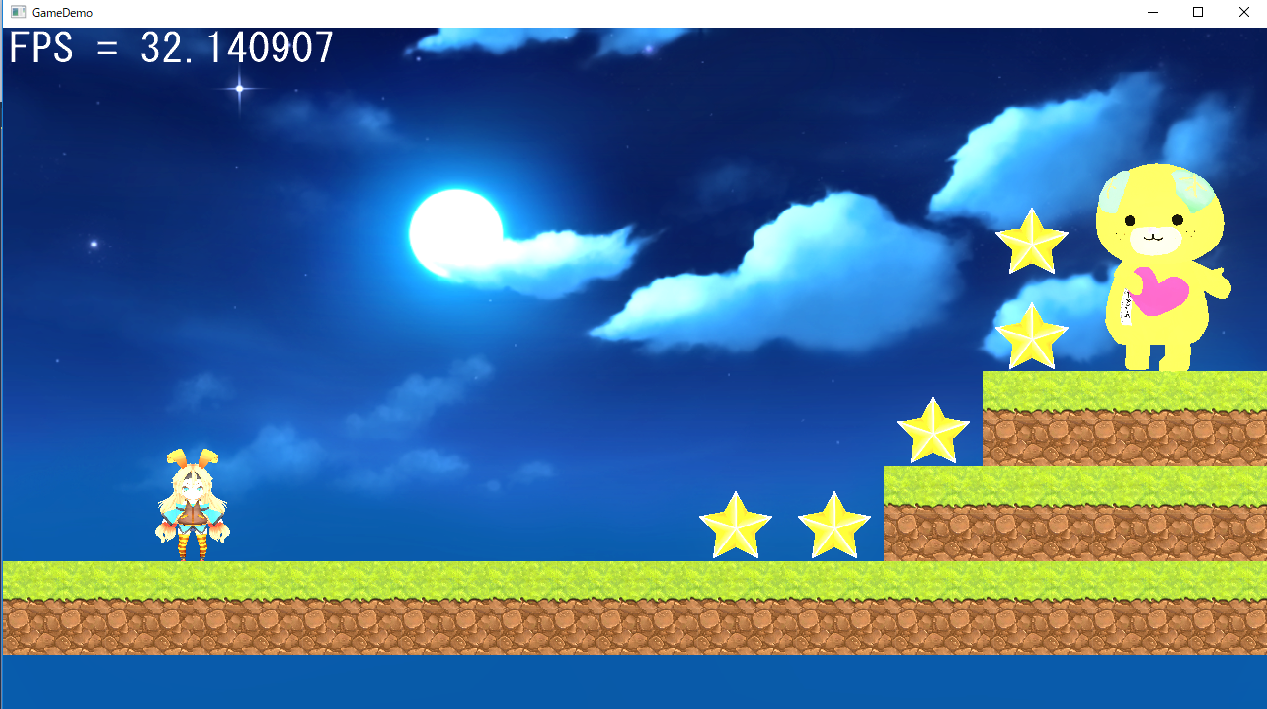
サンプルコード 1-1

|  |
| --- |
| struct Vector{  float x;  　 float y;  };  int main()  {  Vector marioPos;  marioPos.x = 2.5f;  marioPos.y = 2.0f;  　 //ゲームループ  while(true){  //  }  } |

## **1.2 実習課題**

　GameMath\_1/Chapter\_1/Game.cppの27行目のコメントを読んで、図1-4のような絵が完成するように、各オブジェクトの座標を指定しなさい。

図1-4



# **Chapter 2 ベクトル**

## **2.1 ベクトル　数学的な定義**

　ベクトルは数学的には、ただの数字の配列です。この説明を聞いてもイメージはわかないと思いますが、気にしなくて構いません。

## **2.1.1　ベクトルとスカラー**

　数学者はベクトルとスカラーを区別します。スカラーは普通の数字を表す用語です。皆さんが慣れ親しんでいる数字のことです。今後スカラーというキーワードができてたら、ベクトルじゃなく、普通の数字のことなんだなと思ってください。

## **2.1.2　ベクトルの次元**

ベクトルの次元は、そのベクトルにいくつの数が含まれているのかを指します。主にゲームでは2Ｄ、3Ｄ、そして4Ｄベクトル(後半で)を扱います。

## **2.1.3 ベクトルの数学的な記法**

数学では、ベクトルを下記のように記述します。

もしくは

水平に記述するのは**行ベクトル**、垂直に記述するのは**列ベクトル**です。この違いが意味を持つ場合があるのですが、今は同じものだと思っていて構いません。また、ベクトルの各要素はx，y，z，wで表します。x，yで2D、x，y，zで3D、x，y，z，wで4Dです。4Ｄベクトルはアルファベット順でないことに注意してください。4番目の値はwです。

## **2.2　ベクトル　幾何学的な定義**

　ベクトルは幾何学的には大きさと向きを表す線分です。

・ベクトルの大きさは、ベクトルの長さです。そしてベクトルの長さはスカラーです。

・ベクトルの向きはベクトルが空間内でどこを指しているのかを表します。

上のように表記されているベクトルvはv.x = 2、v.y = 4となります。

## **2.2.1 ベクトルはどのように見えるか？**

　2.2のベクトルvは下記のように図2-1のように図示化することができます。



図2-1

## **2.2.2 座標とベクトル**

さて、Chapter1で勉強した、デカルト座標系での位置を表す座標もベクトルを使って表していましたが、実際には座標とベクトルは全くの別物です。

座標は空間の一点を示し、その場所が変わることはありません。しかし、ベクトルは位置情報を持っておらず、大きさと向きのみを保持しています。

点Ｐ＝( ２　４ )　と　ベクトルV = [ 2　４ ]を図示化した図2-2を見てください。



図2-2

このように、赤い矢印はすべて x方向に+2、y方向に+4という大きさを持っているベクトルVとなります。一方点Ｐは赤い丸の一点しか表しません。

## **2.2.3 問題**

問１

　下記の座標P0～P3とベクトルV0～V3を図示化しなさい。

P0 = ( 0　４ )、P1 = ( 1　2 )、P2 = ( －2　6 )、P3 = ( －５　－５ )

V0 = [ 1　2 ]、V1 = [ 2　2 ]、V2 = [ －4　3 ]、V3 = [ －2　－１ ]

Y



X

問２

下記の図に記載されているベクトルを求めなさい。



解答欄

a = [ ] b = [ ] c = [ ] d = [ ]

e = [ ] f = [ ] g = [ ]

# **Chapter 3 ベクトルの演算**

　このチャプターではいくつかのベクトルの演算を学び、その演算の幾何学的な意味を考えていこうと思います。

## **3.1　線形代数 vs 我々が求めるもの**

　主にベクトルを扱う数学の分野を線形代数と呼びます。2.1節で述べたように、線形代数において、ベクトルは数字の配列にすぎません。しかし、我々、ゲームプログラマはベクトルの幾何学的な解釈を求めています。線形代数を扱っている教科書では、幾何学的解釈までは十分に扱っていません。この授業では、ベクトルの幾何学的な解釈に焦点を当てて考えていきます。

## **3.1 ベクトルの反転**

　ベクトルの反転は、ベクトルのすべての要素に―１を乗算することで求まります。例えば

あるベクトルV[ 2　５ ]を反転させると、[ －２　－５　]となります。

## **3.1.1 線形代数の公式**

任意の次元のベクトルを反転するには、単純にベクトルのそれぞれの要素の正負を反転するだけです。例えば、３Ｄのベクトル[　ｘ　ｙ　]を反転させた場合、[　－ｘ　－ｙ　]となります。

***式　－[　ｘ　ｙ　]　＝　[　－ｘ　－ｙ　]***

## **3.1.2 幾何学的解釈**

では、ベクトルの反転の幾何学的な意味を見ていきましょう。ベクトルは反転させると、元のベクトルと真逆のベクトルとなります。下記の図3-1を見てみてください。ベクトルVA[ ２　５　]を反転させたベクトルVBは[　－２　－５　]はVAと真逆を向いていることが分かります。



**VB**

**VA**

図3-1

　ベクトルは位置情報を持たずに、方向と大きさのみを表している数字だったことを思い出してください。図3-1のベクトルVAとVBは完全に真逆のベクトルとなっています。

ベクトルを反転させる計算は、ベクトルに対して―１乗算することで求めることができます。ベクトルの乗算に関しては、後程詳しく見ていきます。

## **3.1.3 問題**

問１

下記のベクトルVA～VDを反転したベクトルVA´～VD´を図示化しなさい。

VA[ ３　２ ]　VB[ 1 ７ ] VC[ －２　５　]　VD[　－３　－２　]



## **3.2 ベクトルの大きさ(長さ)**

これまで、見てきたように、ベクトルは大きさと向きを持っています。しかし、ベクトルの中に大きさも向きもはっきりと表されていないことに気が付いたかもしれません。例えば、ベクトル[　３　４　]の大きさは３でも４でもなく、５です。ベクトルの大きさははっきりと表されていないため、計算しなくてはなりません。

## **3.2.1　線形代数の公式**

　線形代数では、ベクトルの大きさはベクトルを挟む２重の垂直な線を用いて記述します。３次元のベクトルＶの大きさを求める式は次のようになります。

**||Ｖ||＝**

このように、ベクトルの大きさはベクトルの要素の２乗和の平方根となります。

また、ベクトルの大きさは必ず正になります。例えば、ベクトル[　－２　５　－８　]の大きさは下記のように計算されます。

## **3.2.2 幾何学的解釈**

　3.2.1の公式についての理解を幾何学的解釈から深めていきましょう。ベクトルはどんなベクトルであっても、ベクトルvを斜辺として下記の図3-2のように、直角三角形を作ることができます。これは2Dも3Dも同じです。



**|V.ｘ|**

**|V.y|**

**||Ｖ||**

図3-2

ベクトルの大きさ(長さ)の幾何学的な意味は、ベクトルの矢印の長さです。また、図3-2のように、ベクトルｖを斜辺とした直角三角形を作ることができます。また、ピタゴラスの定理より、直角三角形の斜辺の長さは、**残る２辺の長さの２乗和の平方根**となることが分かっているため、次の式が得られます。

・・・・・・・・２Ｄベクトルの場合

　・・・・３Ｄベクトルの場合

これは、まさに3.2.1で見た線形代数の公式と同じになります。

tips

|  |
| --- |
| ベクトルの長さは、２点間の距離の計算を行うときなどに使われる演算で、ゲームで非常によく使われる演算です。２点間の距離の計算を理解するには、ベクトルの減算も学ぶ必要があります。 |

## **3.2.3 問題**

問１

下記のベクトルa~gの長さを小数点第２位までで求めなさい。また、電卓を使っていいものとする。

解答例　a = 10.25



a =

b =

c =

d =

e =

f =

g =

電卓の使い方の動画

<https://www.youtube.com/watch?v=vhPbA0E-8FA&feature=youtu.be>

## **3.3　ベクトルとスカラーの割り算と掛け算**

　ベクトルとスカラーを足すことはできませんが、ベクトルとスカラーの割り算と掛け算はできます。結果は、元のベクトルと平行なベクトルになります。ただし、長さが変わったり、向きが反対だったりします。

## **3.3.1 線形代数の公式**

　ベクトルとスカラーの掛け算と割り算は簡単です。ベクトルの要素それぞれにスカラーを掛けるor割るだけです。スカラーＫと３ＤベクトルＶの掛け算は下記のようになります。

**[　V.x V.y V.z ]×Ｋ**

**＝[　KV.x　KV.y KV.z ]**

例をいくつかあげます。

[ 1　2　3　]×２　＝　[ 2　4　6　]

[ －5　0　0.4　]×－３ = [ 15 0 －1.2 ]

続いて、割り算を見ていきましょう。スカラーＫと3DベクトルVの割り算は下記のようになります。

**[　V.x V.y V.z ]÷Ｋ**

**＝[　V.x/K　V.y/K kV.z/Ｋ ]**

例をいくつかあげます。

[ 4.7 －6 8 ] ÷ 2　=　[ 2.35 －３ 8 ]

[ 10 －30 45 ] ÷ 5　=　[ 2 －6 9 ]

## **3.3.2　幾何学的解釈**

　幾何学的には、ベクトルにスカラーＫを掛け算or割り算するのは、ベクトルの長さをK倍スケーリングする効果を持ちます。(割り算の場合は1/K倍)

例えば、ベクトルの長さを２倍にするには、ベクトルに２をかけます。K＜０なら、ベクトルの向きは反転します。図3-3はベクトルV[ 2 4 ]にいくつかの異なるスカラーを掛けたベクトルを図示化しています。



**－2V**

**V/2**

**2V**

**V**

図3-3

## **3.3.3 問題**

問１

下記のベクトルVA～VDにスカラー３を掛け、ベクトルVA´～VD´を図示化しなさい。

VA[ ３　２ ]　VB[ 1 2 ] VC[ －２　4　]　VD[　－３　－２　]



問2

下記のベクトルVA～VDをスカラ2で割った、ベクトルVA´～VD´を図示化しなさい。

VA[ 4　２ ]　VB[ 6 4 ] VC[ －２　4　]　VD[　－4　－4　]

