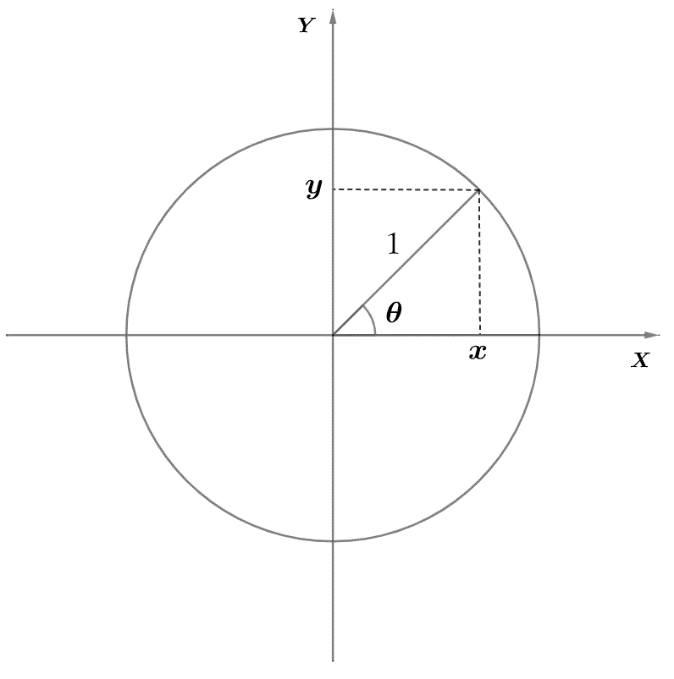
三角関数②

・そのあと、ここの角度が０＜θ＜９０°しか使えなかったので、

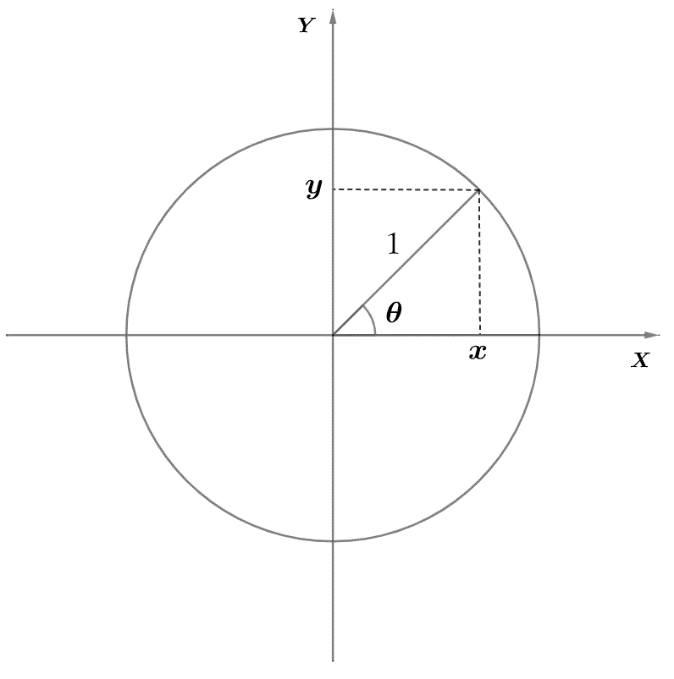
単位円で拡張して、θはいろんな角度をつかえるようになりました。

＋の反時計回りとーの時計回りもOKでしたね。

～半径１の円（単位円）による**定義**～

はい、先週はこんな感じでしたが、

何か質問のし忘れとかありませんか？

**～単位円の方程式～**

単位円の方程式は次式で表されます。

単位円の方程式、ちょっと考えてみて。

作業＆質問タイム

プリントにヒントがあるから。

では正解発表。

なんでこの方程式になるかわかりますか？

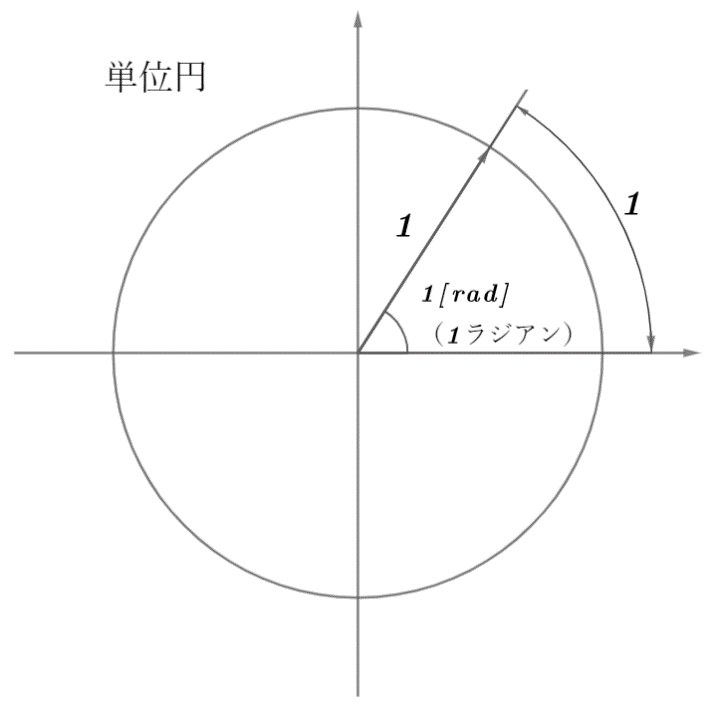
みんなすでに答え知ってるはずだよ。三平方の定理。

はい、中心から距離が１のところのｘ、ｙの組を

全部点打っていったら円になるよね。

距離なんで、だよね、

これ、両辺を２乗すると **になるよね。**

■度じゃないよ！　ラジアンよ！

　本日のポイント

「弧」の長さが、半径と同じ長さになる角度のことを　　ラジアン（**）**と呼びます。

このような角度の測り方を　**弧度法**と呼びます。

**例題**

単位円の一周の長さは（　　　　）ですよね。

ということは

※慣例的に、弧度法の場合は「ラジアン」は省略します（読み、書き共に）。

■利点本題（sin, cos、マクローリン展開）

はい、ここからが本題、

みなさん、C言語とかC#のプログラムで sinとかcos使ったことありますか？

ではその中身が実際にどんな計算していると思う？どう？

三角関数表？小数点以下の角度はどうしますか？

どのくらいの桁まで持てばいいの？とかなりそうだよね。たくさん持つとメモリたくさん必要になりそうだよね。

はい、実はこんな感じで、精度が高い近似値を使って計算しています。

マクローリン展開

たとえば　sin　の近似計算を度数法（～度）でおこなうと

というように式がややこしいし、何より計算する量が多くなるよね、

とくに乗数がどんどん増えていくからね。

ちなみに！は階乗っていって、３！だと１ｘ２ｘ３、５！だと１ｘ２ｘ３ｘ４ｘ５。

それではここで弧度法をつかって近似するとどうなるでしょうか？

１８０度って弧度法だと何になるんだっけ？　そうπだよね。

ということはここが全部１になるよね。整理すると

こういう感じになってすっきりしたね。

できるだけプログラムの処理は軽くしたいのでこっちのほうをライブラリでは使ってます。

厳密にはライブラリごとに、これ以外の計算を色々工夫してるみたいです。

～ラジアンの利点～

ゲームプログラムではライブラリ関数のsin()、cos()などの三角関数を多用する場合があります（エフェクト、キャラやカメラの動きなど）。コンピュータはsin()、cos()関数の値をどのように求めているのでしょうか？

マクローリン展開（…興味がある人は調べてみてください）という近似計算をして値を求めています。

たとえばsin()関数を見てみましょう。

度数法による近似計算(マクローリン展開)

弧度法による近似計算(マクローリン展開)

弧度法を使えば、180°=πとなり、　 となって度数法による式を簡単にすることができます。

※　マクローリン展開は覚えなくていいです。

※　そんなものがある、という程度でOK。

　本日のポイント

このように弧度法を使うことで

計算量を減らすことができて、プログラムの処理速度を改善することができます。

**プレイヤの入力が、すぐさまキャラクタやゲームの内容に反映されることが望まれるゲーム（格ゲー、シューティング、アクション、FPSなど）では、高いフレームレートが必要になります。フレームレートを上げるには、１フレームあたりの処理時間を減らす必要があります。**

また、３Dキャラクタのアニメーションにはたくさんの回転計算が含まれていて、回転には三角関数が使われています。ハック＆スラッシュゲームや軍勢アクションゲームなど、たくさんのキャラクタが動くゲームでは、計算量を減らすことで、さらにたくさんのキャラクタやエフェクトを表示させることができたり、より豪華（派手な）表現をさせることが可能になります。

はい、今日はまずですね、

三角関数の角度に関係した性質を見ていきたいと思います。

■三角関数の角度による性質

性質①　角度が

*※N周回って同じところ*

下の性質②～⑥について、単位円を描いて考えてみよう。

（

性質②　角度が

性質③　角度が

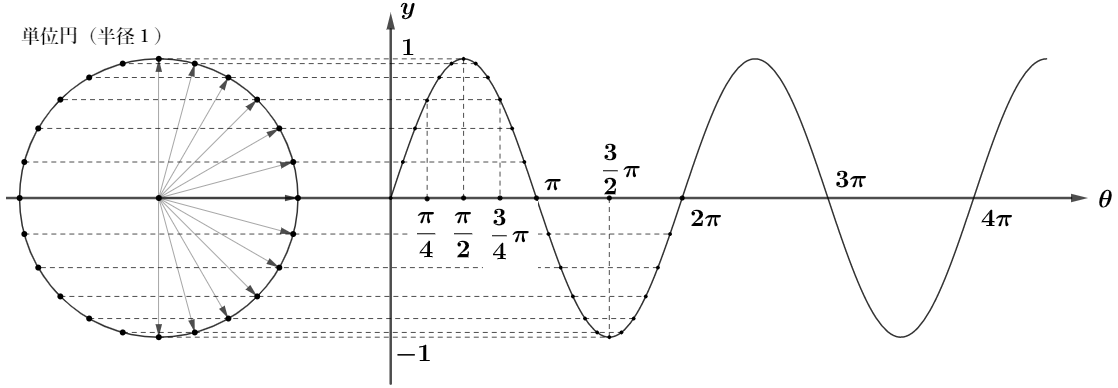
性質④　角度が

性質⑤　角度が

性質⑥　角度が

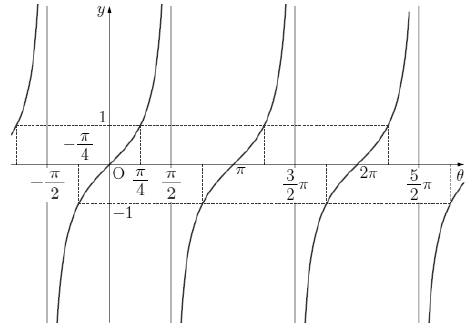
つぎはですね。最初の授業で水面の波を三角関数を使って、シェーダーで作れる話をしたと思うんですが、

三角関数がなぜ波と関係しているのかを、グラフを描いて見てみたいと思います。

■三角関数のグラフ

**を描いてみよう**

**を描いてみよう**

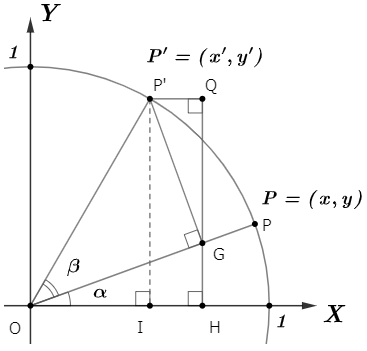


**を描いてみよう**

■グループワーク



ヒント　　下図の点のY座標を求めることと同じ

※持っている道具は、角αとβ、単位円、相似の関係、そして三角関数



ヒント　　上図の点のX座標を求めることと同じ

※上図から探してもいいし、「角度による性質」を使って変形してもOK

■原点を中心とした任意の点の回転

前のページの　①、　②　を（　　　**加　　法　　定　　理**）と呼びます。

ではPの座標を(x, y)としたときに、それぞれsinとcosを使ってどう書けるでしょうか？

の座標をとすると、角αについての三角関数を使って、次のように書けます。

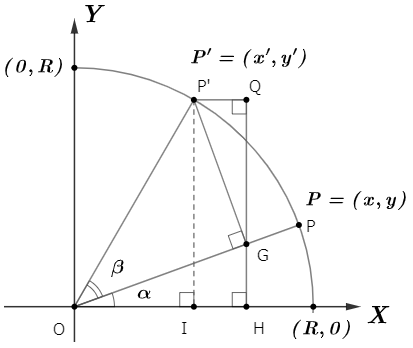
したがって、前のページの式　①、　②　は

となり、次のことが言えます。  
  
回転半径が　　の任意の点を、原点を中心にθだけ回転させた点は

次式で求められます。

■グループワーク（任意の点の回転）

回転半径　　を任意の回転半径*R*に拡張してください。

点の座標を点の座標を使った式で書いてください。

本日のポイント

XY平面上の任意の点P（すべての座標）は、

原点からの距離*R*とX軸からの回転θで表現できるので、つぎのことが言えます。



XY平面上の任意の点を、原点を中心にθだけ回転させた点は

次式で求められます。

■グループワーク（多関節の制御）　　… 本日のポイント

【前提条件】

点Jは原点Oを中心とした回転だけで動くものとします。

また、点Pは点Jを原点とした回転だけで動くものとします。

【問題】

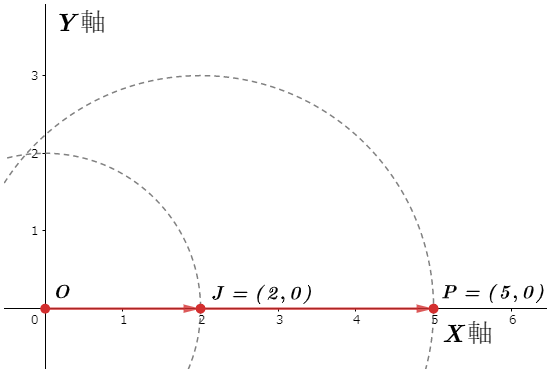
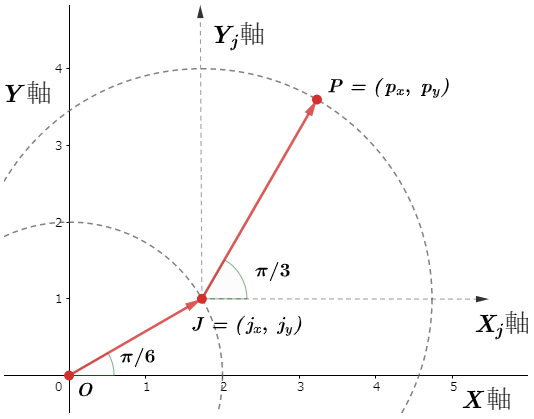
いま、下図の「初期状態」から、点Jを　ラジアン回転、点Pを　ラジアン回転させた状態を

「変化後状態」とします。「変化後状態」の点J、および点Pの座標を求めてください。

ヒント　　「初期状態」の点Pの座標は原点Oからの座標。

点Jと点Pについて、それぞれの原点の座標系で考える。

【変化後状態】



【初期状態】