**Stavy hry**

Nyní nastává ta nejtěžší část, musíme zapřemýšlet do budoucnosti a rozhodnout se, jaké stavy naše hra bude mít. Můžeme dojít k následujícím třem stavům:

* výběr, jestli má být druhý hráč ovládán hráčem nebo počítačem
* začátek kola, kdy je hra pozastavena, aby měli hráči možnost se nachystat
* samotné kolo, ve kterém se oba hráči pohybují a létá mezi nimi míč

Jak tyto stavy reprezentovat? Nejsnazší asi bude každý stav napsat jako metodu, přičemž budeme mít proměnnou, ve které bude uložen odkaz na aktuální stav, který se bude volat. Nazvěme si tedy stavy například následovně:

* logic\_game\_start()
* logic\_round\_start()
* logic\_game\_body()

Dále si vytvoříme nový atribut logic (jako logika aktuálního stavu), který pak na začátku funkce main() nastavíme na první stav:

Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.

Když se ještě zamyslíme, můžeme dojít k závěru, že některé objekty (konkretně oba hráče, míč a skóre) budeme chtít mít vykreslené úplně pokaždé, takže je nemusíme členit do jednotlivých logik a můžeme je přidat přímo do cyklu v main(). Všechny naše objekty budou bílé a skóre bude nahoře uprostřed:

Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.Tím bychom dokončili náš hlavní herní cyklus, takže nám zbývá už pouze vytvořit jednotlivé stavy a máme hotovo!

**Začátek hry**

Začněme s prvním stavem, tedy se začátkem hry. V tomto stavu po uživateli chceme, aby se rozhodl, jestli chce hrát proti druhému člověku, nebo počítači. Dejme tomu, že ho vyzveme, ať stiskne H (*human*), pokud bude chtít hrát proti člověku, nebo C (*computer*), pokud bude chtít hrát proti počítači. Naši otázku můžeme umístit doprostřed obrazovky a podle odpovědi uživatele nastavíme proměnnou player2\_ai:

Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.**Začátek kola**

Před začátkem každého kola je dobrý nápad umístit hráče a míč do původní pozice a vyresetovat míči směr a rychlost. Pak nám již stačí pouze hráče vyzvat, aby stiskl Mezerník, čímž signalizuje, že jsou všichni připraveni a hra může začít:

Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.**Samotné kolo**

Jak jste správně vytušili, toto je nejtěžší část celého programu. Právě v ní se totiž odehrává drtivá většina všech aktivit:

Obsah obrázku text, Písmo, snímek obrazovky, Grafika

Obsah vygenerovaný umělou inteligencí může být nesprávný.

**Pohyby hráčů**

Začneme tím, že se vypořádáme s pohyby hráčů. Řekněme, že hráč číslo 1 bude pro pohyb nahoru používat W a pro pohyb dolů S. Pohybovat se bude samozřejmě moci pouze v případě, že nenaráží do vrchní nebo spodní části obrazovky:

Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.**AI počítače**

Pro případ, že je druhý hráč ovládán počítačem, vytvoříme jednoduchou dvouřádkovou AI. Ta se bude pokoušet být na stejné výškové hladině, jako je míč. Pohyb vypočítáme jednoduše:

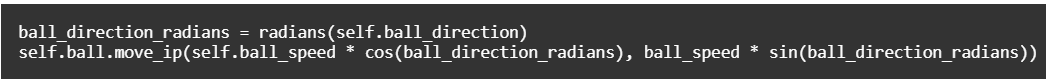
Pokud je míč níže než hráč, musí jít hráč níže (tj. posun hráče se musí pohybovat v kladných číslech). Pokud je ale míč výše, musí se hráč posunout o záporné číslo. Hráč se v jednom kroku herního cyklu nikdy nemůže posunout dále, než je jeho rychlost. Zároveň, jelikož nechce míč přeletět, posune se maximálně tolik, aby se přesně zarovnal s míčem. Vzdálenost pohybu tak vypočítáme pomocí min(player\_speed, abs(player2.centery - ball.centery)).

Abychom určili, jestli se má hráč pohybovat v kladných, nebo záporných číslech, potřebujeme funkci, která nám zjistí znaménko rozdílů souřadnic míče a hráče. Za tímto účelem si vytvoříme funkci sign:

Obsah obrázku text, snímek obrazovky, Písmo, řada/pruh

Obsah vygenerovaný umělou inteligencí může být nesprávný.**Pohyb míče**

Pohyby hráčů již máme tedy vyřešené. Teď si ještě vyřešíme pohyb míče, který přímo vyplývá z [jednotkové kružnice](https://cs.wikipedia.org/wiki/Jednotkov%C3%A1_kru%C5%BEnice). Použijeme funkce sin() a cos() pro získání nové pozice míče podle směru, kterým zrovna letí a jeho rychlosti:

**Odrážení**

Když si nyní spustíme naši hru, tak se nám nebude příliš dařit, protože, i přes všechnu naši snahu, míč vždy opustí hrací plochu. Proto musíme ještě definovat, co má míč dělat při nárazu do hráčů nebo některého okraje.

**Odražení od horní a dolní hrany**

**Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.**

**Odražení od levé a pravé hrany**

Poslední situace, kterou nám ještě zbývá vyřešit, je, když míč narazí do pravé nebo levé strany. V takovém případě to znamená, že jeden z hráčů nebyl schopný míč dostatečně rychle odrazit. Druhému hráči tedy přičteme bod a vrátíme se na stav začátku kola:

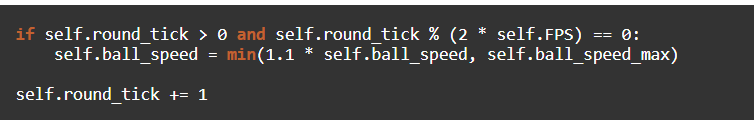
Obsah obrázku text, Písmo, snímek obrazovky, typografie

Obsah vygenerovaný umělou inteligencí může být nesprávný.

**Zrychlování**

A posledním vylepšením, které jsme si slíbili, ale ještě jsme jej nezakomponovali, je samozrychlující se míč. Řekněme, že každé 2 sekundy kola se míč sám zrychlí o 10% až do své maximální rychlosti (lidé mají své limity, kdy míč ještě dokáží sledovat )

Ke zjištění toho, zda-li uplynuly právě 2 sekundy, nám poslouží náš časovač round\_tick, jehož hodnota se každou iteraci v kole zvýší o 1. A každé 2 sekundy to jsou právě, když round\_tick je násobkem dvojnásobku FPS:



https://www.itnetwork.cz/python/pygame/pygame-pong-logika-stavu-hry-a-dokonceni