



Lekcja 4. Moby i przedmioty

Cel lekcji

Celem zajęć będzie poznanie mechanizmów odpowiadających za pojawianie się różnego rodzaju mobów oraz przedmiotów w grze z wykorzystaniem pętli.

Agenda

1. Quiz
2. Pytania wstępne i rozmowa
3. Pętla for
4. Zombie run
5. Pajęczne siedlisko

QUIZ

Uczestnicy wykonują quiz, do którego mają dostęp z poziomu **panelu ucznia** (zakładka *Dashboard* lub *Moje Kursy*).

Quiz do wglądu dla trenera:

<https://quiz.giganciprogramowania.edu.pl/python-mc-q1>

Powyższy link możemy udostępnić uczestnikom TYLKO jeśli mają problemy z zalogowaniem się do panelu ucznia. Wyniki wtedy dla tych osób musimy ręcznie zapisać w liście obecności (przycisk *Info* przy danym uczestniku).

Zapoznaj się z treścią quizu przed zajęciami.

Pamiętaj żeby po zakończeniu quizu omówić pytania i odpowiedzi.

Pytania wstępne

1. Co to są funkcje?
2. Jak utworzyć własną funkcję?
3. Jak budować z użyciem czatu?

Kompilator Online

<https://www.online-python.com/>

Pętla for:

Kolejny element programowania jaki poznamy to pętla. Pętla jest to taki element języka, który pozwala powtarzać kilkakrotnie pewne fragmenty skryptów. Na dzisiejszych zajęciach poznamy jedną z pętli, które występują w języku Python – **for**. Wyobraźmy sobie, że ktoś poprosił nas o napisanie skryptu, który wyświetli liczby od 0 do 100. Teoretycznie moglibyśmy napisać sto jeden razy funkcje **print()**, ale nie jest to raczej wygodny sposób. Możemy do tego wykorzystać wspomnianą pętlę **for**:

```
for i in range(101):  
    print(i)
```

Opowiadamy, że **range()** jest to funkcja która zwraca kolejne liczby z podanego zakresu, domyślnie zaczynając od zera i w każdym wykonaniu pętli do zmiennej pętli **i** przypisywana jest ta liczba. Liczba podawana jako górna część zakresu nie jest do niego uwzględniona.

Do funkcji **range()** możemy przekazać dwa parametry (oddzielone przecinkiem), wtedy określamy zakres liczb od do:

```
for i in range(5, 10):  
    print(i) # wynik: 5, 6, 7, 8, 9
```

Zwrócimy uwagę, że podany zakres 5 – 10, wylistuje liczby od 5 do 9.

Istnieje też wersja funkcji **range()** z trzema parametrami, wtedy trzeci parametr określa krok (czyli o ile ma być zwiększana wartość zmiennej w każdej iteracji). Na przykład wyświetlenie tylko liczb nieparzystych:

```
for i in range(1, 10, 2):  
    print(i)  
  
# wynik: 1, 3, 5, 7, 9
```

Zobaczmy zastosowanie pętli w bardziej rozbudowanych skrypcie. Napiszemy skrypt, który będzie w stanie wyświetlić na ekranie sumę wszystkich liczb podanych przez użytkownika. Na początku program ma pytać, ile liczb chcemy zsumować, a następnie prosi kolejno o ich wpisanie. Na końcu wyświetla ich sumę.

```
ilosc = int(input("Podaj ile liczb chcesz zsumować? "))  
  
# tworzymy zmienną na sumę  
suma = 0  
  
for i in range(1, ilosc + 1):
```

```

    liczba = int(input("Podaj liczbę " + str(i) + ": ")) # pobieramy
kolejną liczbę do dodania
    suma = suma + liczba

    # do zmiennej suma za każdym razem dodajemy to co w tej zmiennej
już jest plus nowa podana liczba
    # po zakończeniu pętli wyświetlamy wynik
print("Suma podanych liczb: " + str(suma))

```

MINECRAFT

ZOMBIE RUN

Tworzymy potwory w dużej ilości wokół gracza. Naszym zadaniem jest ucieczka aż do momentu, w którym zombie spłoną. Pozostałe potwory dobijamy otrzymanym toporem.

<https://minecraft-pl.gamepedia.com/Zombie>



```

def zombie_spawn():
    #tryb przetrwania
    gameplay.set_game_mode(SURVIVAL, mobs.target(NEAREST_PLAYER))
    mobs.give(mobs.target(NEAREST_PLAYER), DIAMOND_AXE, 1)
    #używamy pętli, aby utworzyć potwory
    for i in range(100):
        mobs.spawn(ZOMBIE, randpos(pos(-30, 0, -30), pos(30, 0, 30)))
    )

player.on_chat("zombie",zombie_spawn)

```



AI w zachowaniu mobów

UWAGA: Można poruszyć ten temat na początku rozmowy o dodawaniu Zombie do kodu - należy dostosować ten moment do wiedzy uczestników np. jeśli mamy osoby, które nigdy nie grały w MC poza zajęciami należy najpierw wykonać kod, a następnie przeprowadzić rozmowę.

Pytamy:

- Czy uczniowie słyszeli kiedykolwiek o AI (SI) w grach? [np. AI używane jest do sterowania mobami]
- Czy uczestnicy zauważyli jak zombie reagują na gracza oraz czy widzą go zawsze? [Widzą z pewnej odległości i atakują jeśli go zobaczą]
- Czy wiedzą jak zachowują się zombie, które płoną w normalnej grze? [jeśli nie atakują gracza, a obok jest jakieś drzewo to wejdą pod nie. Mogą też wejść do wody by się ugasić]

Dzięki AI mobów, czyli pewnych zasad, które mówią im jak mają się zachować w danej sytuacji, potwory i moby neutralne lub przyjazne ożywiają świat Minecrafta. Taka sztuczna inteligencja (SI po polsku) lub Artificial Intelligence (AI po angielsku) jest dość prosta, ale i tak daje radę stworzyć świetną grę.

Najprostszym przykładem działania pod jakimś warunkiem jest użycie komendy na czacie: Jeśli gracz napisze słowo, które wcześniej ustalił w kodzie - wtedy kod wykona się. Jest to uproszczony sposób tego jak działa proste AI w grach.

Temat tego jak dokładnie działają pewne aspekty SI poznamy nieco później, ale już teraz możemy porozmawiać na ten temat dalej podczas ćwiczenia.

Ćwiczenie:

Postaraj się opisać zasady zachowania wybranego moba w komentarzu po naszym kodzie używając zwrotu “jeśli”. To może być pajak - za chwilę stworzymy ich siedlisko, ale również owca, wilk czy dowolny inny, który przyjdzie nam do głowy.
np.

```
# Jeśli wilk zobaczy szkieleta to go zaatakuje  
# Jeśli wilk dostanie kość to może się oswoić  
# Jeśli wilk jest oswojony i ktoś zaatakuje gracza to wilk go obroni  
# Jeśli wilk jest oswojony i gracz kogoś zaatakuje to wilk mu pomoże  
# Jeśli gracz każe wilkowi siedzieć i ten jest oswojony to usiądzie
```

Do tych komentarzy wrócimy za dwa zajęcia.

PAJĘCZE SIEDLIŚKO

https://makecode.com/_D050btFJ2hdx

Mini gra polega na utworzeniu jaskini, w której gracz zostaje uwięziony. W jaskini rozmieścimy bloki pajęczyny. Gracz wchodząc na taki blok wytraca prędkość. Dodatkowo spawnujemy pająki, które trzeba będzie pokonać za pomocą broni otrzymanych w ekwipunku. Dojście do jasnogłazu i zniszczenie go teleportuje gracza w bezpieczne miejsce.

Czyszczenie ekwipunku gracza

Nie ma bezpośredniej funkcji, która czyści ekwipunek, ale można skorzystać z funkcji która wywołuje komendy mc na czacie.

Przykład

```
player.execute("/clear")
```

Tworzenie jaskini-dodatkowy parametr funkcji blocks.fill()

Dotychczas bryły puste w środku tworzyliśmy w taki sposób że najpierw tworzyliśmy jednolitą bryłę a później w jej wnętrzu tworzyliśmy bryłę z bloku powietrza. Teraz wykorzystamy dodatkowy parametr funkcji blocks.fill(), dzięki niemu od razu stworzymy jaskinię.

<https://minecraft.makecode.com/reference/blocks/fill>

<https://minecraft-pl.gamepedia.com/Paj%C4%99czyna>

<https://minecraft-pl.gamepedia.com/Paj%C4%85k>



Pobieramy aktualną pozycję gracza

```
pozycja=player.position()
x=pozycja.get_value(Axis.X)
y=pozycja.get_value(Axis.Y)
z=pozycja.get_value(Axis.Z)
```

Tworzymy funkcję której zadaniem będzie utworzyć jaskinię

```
def jaskinia():
    blocks.fill(BEDROCK, world(x-5,y-1,z-1),world(x+10,y+10,z+50),FillOperation.HOLLOW)
    blocks.place(GLOWSTONE, world(x,y,z+49))
    #spider_spawn()

player.on_chat("pająki", jaskinia)
```

Definicja funkcji spiderSpawn()

```
def spider_spawn():  
  
    gameplay.set_game_mode(SURVIVAL, mobs.target(NEAREST_PLAYER))  
    #czyszczenie ekwipunku  
    player.execute("/clear")  
    mobs.give(mobs.target(NEAREST_PLAYER), DIAMOND_SWORD, 1)  
    mobs.give(mobs.target(NEAREST_PLAYER), TORCH, 3)  
    #używamy pętli aby utworzyć pajęczyny  
    for i in range(50):  
  
        blocks.place(COBWEB, randpos(world(x-4,y+1,z), world(x+9,y+7  
,z+45)))  
  
        for i in range(5):  
  
            mobs.spawn(SPIDER, randpos(world(x-4,y+1,z+10), world(x+9,y+7,z+45)))
```

Funkcja wywoływana po zniszczeniu konkretnego bloku

```
#zdarzenie na zniszczenie konkretnego bloku  
def zniszczenie_bloku():  
    gameplay.set_game_mode(CREATIVE, mobs.target(NEAREST_PLAYER))  
    player.teleport(pos(50, 1, 50))  
blocks.on_block_broken(GLOWSTONE, zniszczenie_bloku)
```

Uwaga, w powyższym przykładzie koordynaty gracza są pobierane tylko raz po uruchomieniu programu.

Jeżeli chcemy budować ponownie należy ponownie uruchomić program (zieloną strzałką) aby pobrać aktualne współrzędne gracza. Wpisanie tylko komendy “pajaki” bez wcześniejszego ponownego uruchomienia programu wybuduje jaskinie w tym samym miejscu.

(každorazowe otworenie edytora Makecode (przyciskiem c) i zamknięcie przyciskiem X NAWET z pominięciem wciśnięcia

zielonej strzałki RÓWNIEŻ uruchamia projekt, ALE zalecamy używać przycisku zielonej strzałki w celu testowania programu, jest to bardziej intuicyjny sposób).

Opcjonalnie

Wersja powyższego programu, ale korzystającego ze zmiennych globalnych dzięki, którym możemy pobierać współrzędne za każdym razem kiedy wywołujemy komendę “pająki”

https://www.w3schools.com/python/python_variables_global.asp

Tworzymy funkcję do pobierania aktualnej pozycji gracza

```
x, y, z = 0, 0, 0
def pobierz_pozycje_gracza():
    global x, y, z
    pozycja = player.position()
    x = pozycja.get_value(Axis.X)
    y = pozycja.get_value(Axis.Y)
    z = pozycja.get_value(Axis.Z)
```

W funkcji jaskinia wywołujemy nowo utworzoną funkcję

```
def jaskinia():
    pobierz_pozycje_gracza()
    blocks.fill(BEDROCK, world(x-5,y-1,z-1), world(x+10,y+10,z+50),
FillOperation.HOLLOW)
    blocks.place(GLOWSTONE, world(x,y,z+49))
    #spiderSpawn()
player.on_chat("pająki", jaskinia)
```

Całość: https://makecode.com/_30j7Ez91AR8i

Podsumowanie

1. Co to jest pętla i jak działa?
2. Jakie poznaliśmy nowe zdarzenia?
3. Jak działa funkcja range()?

4. Jaka jest różnica pomiędzy pozycją względną gracza a pozycją świata gracza.?