ORIGEM DA LINGUGEM

Van Rossum decidiu criar uma linguagem de programação que tivesse comandos simples, diferentemente da linguagem C.

Guidom Van Rossum

MENU 01 07 12

PRIMEIRO COMANDO

```
1 print('Olá, mundo!')
2 # Função para mostrar algo na tela
```

Resultado

```
Olá, mundo!
```

TIPOS PRIMITIVOS

Strings

As strings são sequências de caracteres. O python permite formar strings com aspas simples ou aspas duplas.

Númericos

Os tipos númericos são divididos em inteiros(int), ponto flutuante(float) e números complexos(complex).

Booleanos

O tipo booleano(bool) é definido como valor True(verdadeiro) ou False(falso).

EXEMPLO!

Número = 5

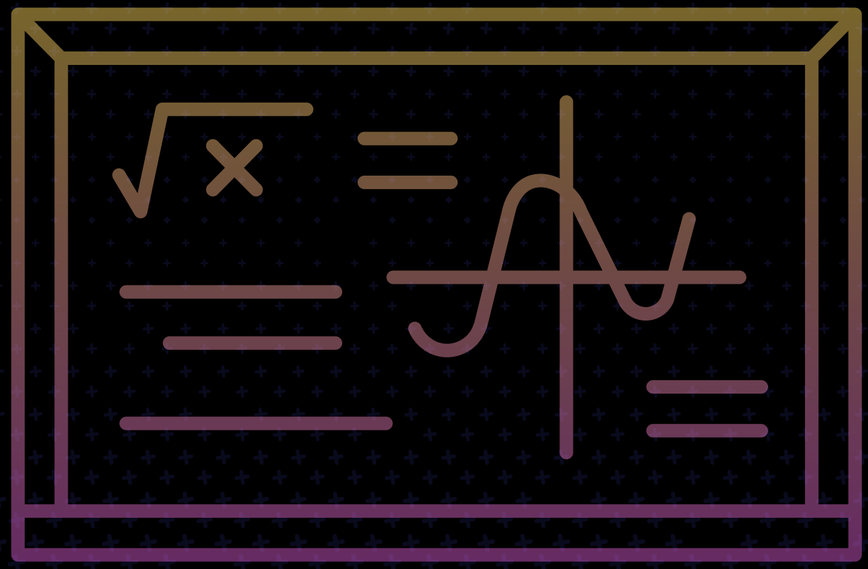
Tipo inteiro (int)

Número = 5.0

Tipo Flutuante (float)

Número = 1j

Tipo complexo (complex)



MENU 01 07 12

LÓGICA!

```
1 print('Olá, mundo!')
2 print(7 + 4)
3 print('7' + '4')
```

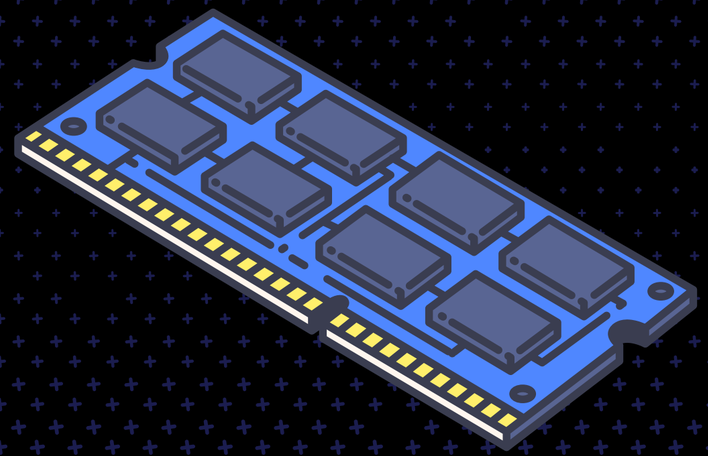
Resultado

```
Olá, mundo!
11
'74'
```

MENU 01 07 12

VARIÁVEIS

Variáveis são espaços na mémoria do computador para referênciar um valor.



MENU 01 07 12

SEGUNDO COMANDO

```
1 mensagem = 'Olá, mundo!'
2 # Variável mensagem RECEBE o valor Olá, mundo!
3 print(mensagem)
```

Resultado

Olá, mundo!

TERCEIRO COMANDO

```
1 nome = str(input('Digite seu nome: ')
2 idade = int(input('Digite a sua idade: '))
3 peso = float(input('Digite o seu peso: '))
4 print(nome, idade, peso)
```

PLAYER 1

HIGHSCORE 2500

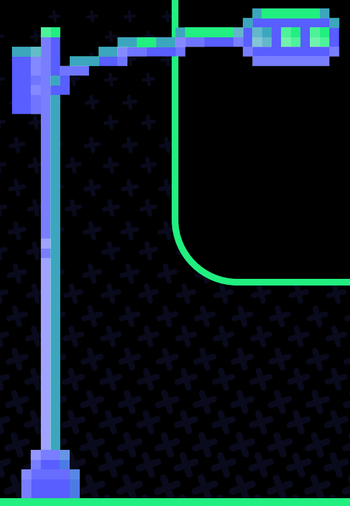
PLAYER 2

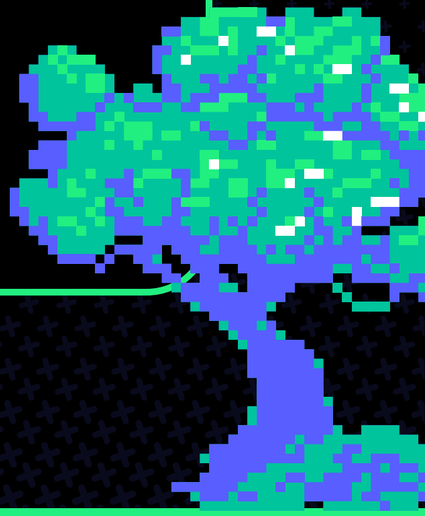
CONCATENAÇÃO

START

MENU

SIGN IN





MENU 01 07 12

UNIÃO DE VALORES

```
1 nome = 'João'
2 idade = 29
3 peso = 88.53
4 print(nome, ' tem', idade, ' anos de idade e pesa', peso, 'Kg.')
```

Resultado

João tem 29 anos de idade e pesa 88.53 Kg.

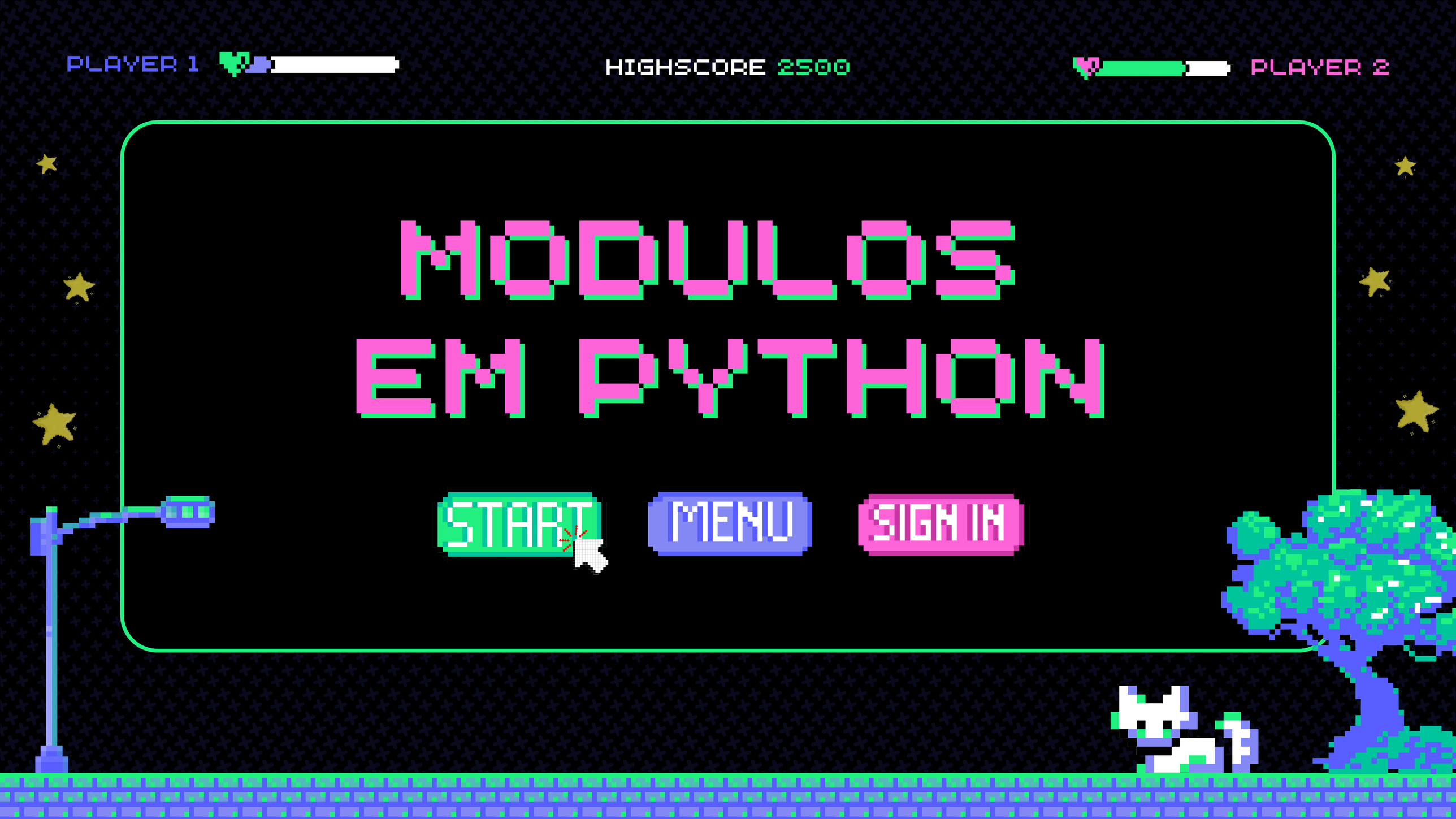
MENU 01 07 12

UNIÃO DE VALORES COM O FORMAT

```
1 nome = 'João'
2 idade = 29
3 peso = 88.53
4 print(f'{nome} tem {idade} anos de idade e pesa {peso}Kg.')
```

Resultado

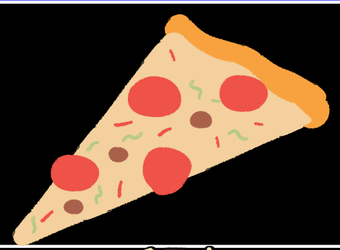
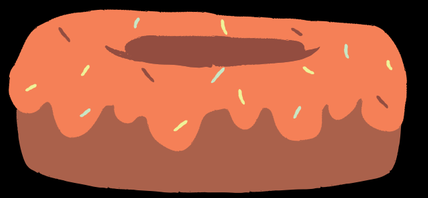
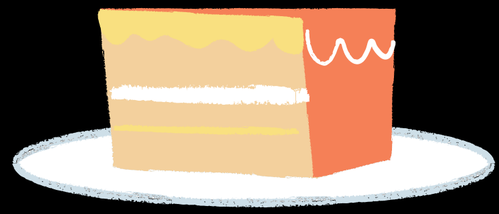
João tem 29 anos de idade e pesa 88.53 Kg.


PLAYER 1
HIGHSCORE 2500
PLAYER 2
MÓDULOS
EM PYTHON
START
MENU
SIGN IN

MENU 01 07 12

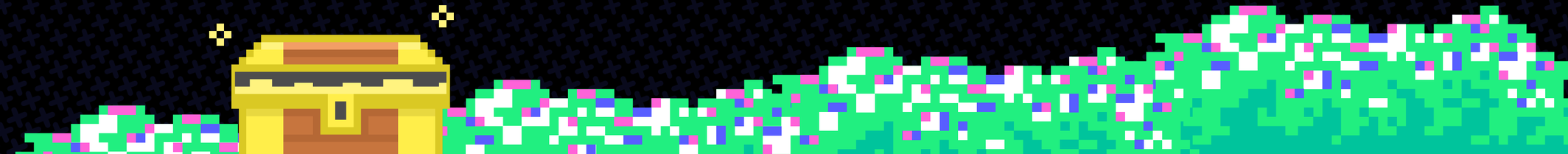
O QUE É UM MODULO

- MODULOS SÃO BIBLIOTECAS QUE COMPLEMENTAM A LINGUAGEM. SERVINDO DE CONTEÚDO ADICIONAL.

BEBIDA	COMIDA	DOCE
		
		
		

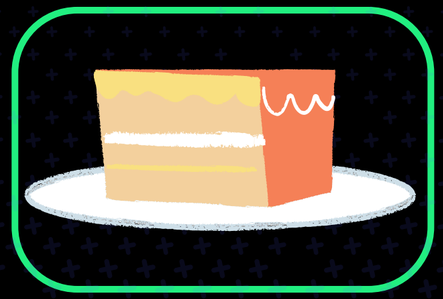
IMPORT BEBIDA

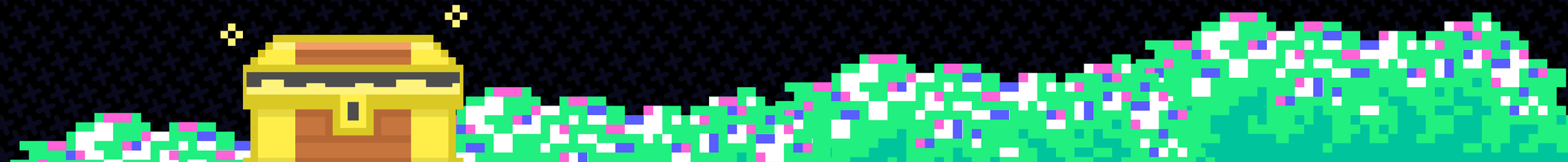
IMPORT COMIDA



```
FROM BEBIDA IMPORT SUCO
```

```
FROM DOCE IMPORT BOLO
```





BIBLIOTECA MATH

```
FROM MATH IMPORT SQRT
```

SQRT

```
FROM MATH IMPORT POW
```

POW

```
IMPORT MATH
```

CEIL

SQRT

FLOOR

FACTORIAL

TRUNC

POW

PLAYER 1

HIGHSCORE 2500

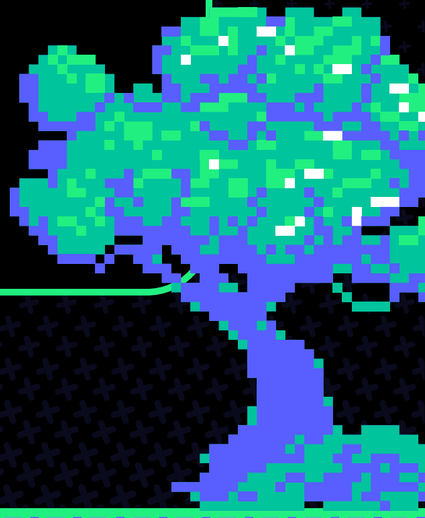
PLAYER 2

OPERADORES

START

MENU

SIGN IN



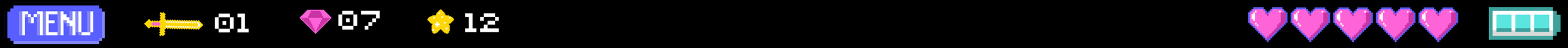
MENU 01 07 12

OPERADORES DE ATRIBUIÇÃO

operador	exemplo	equivalente a
=	x = 1	x = 1
=+	x += 1	x = x + 1
-=	x -= 1	x = x- 1
*=	x *= 1	x = x * 1
/=	x /= 1	x = x / 1
%=	x %= 1	x = x % 1

OPERADORES DE COMPARAÇÃO

operador	conceito	exemplo
>(maior)	verifica se um valor é maior que outro	x > 1
<(menor)	verifica se um valor é menor que outro	x < 1
>=(maior ou igual)	verifica se um valor é maior ou igual que outro	x >= 1
<=(menor ou igual)	verifica se um valor é menor ou igual que outro	x <= 1
==(igual)	verifica se um valor é igual que outtro	x == 1
!=(diferente)	verifica se um valor é diferente de outro	x != 1



OPERADORES LÓGICOS

operador	conceito	**exemplo**
AND	x = 1	x > 1 and x < 5
OR	x += 1	x > 1 or x < 5
NOT	x -= 1	not(x > 1 and x < 5)

OPERADORES DE IDENTIDADE

operador	conceito	**exemplo**
IS	retorna True se as variáveis comparadas forem o mesmo objeto	nome is 'marcos'
IS NOT	retorna True se as variáveis comparadas não forem o mesmo objeto	x is not 'python'

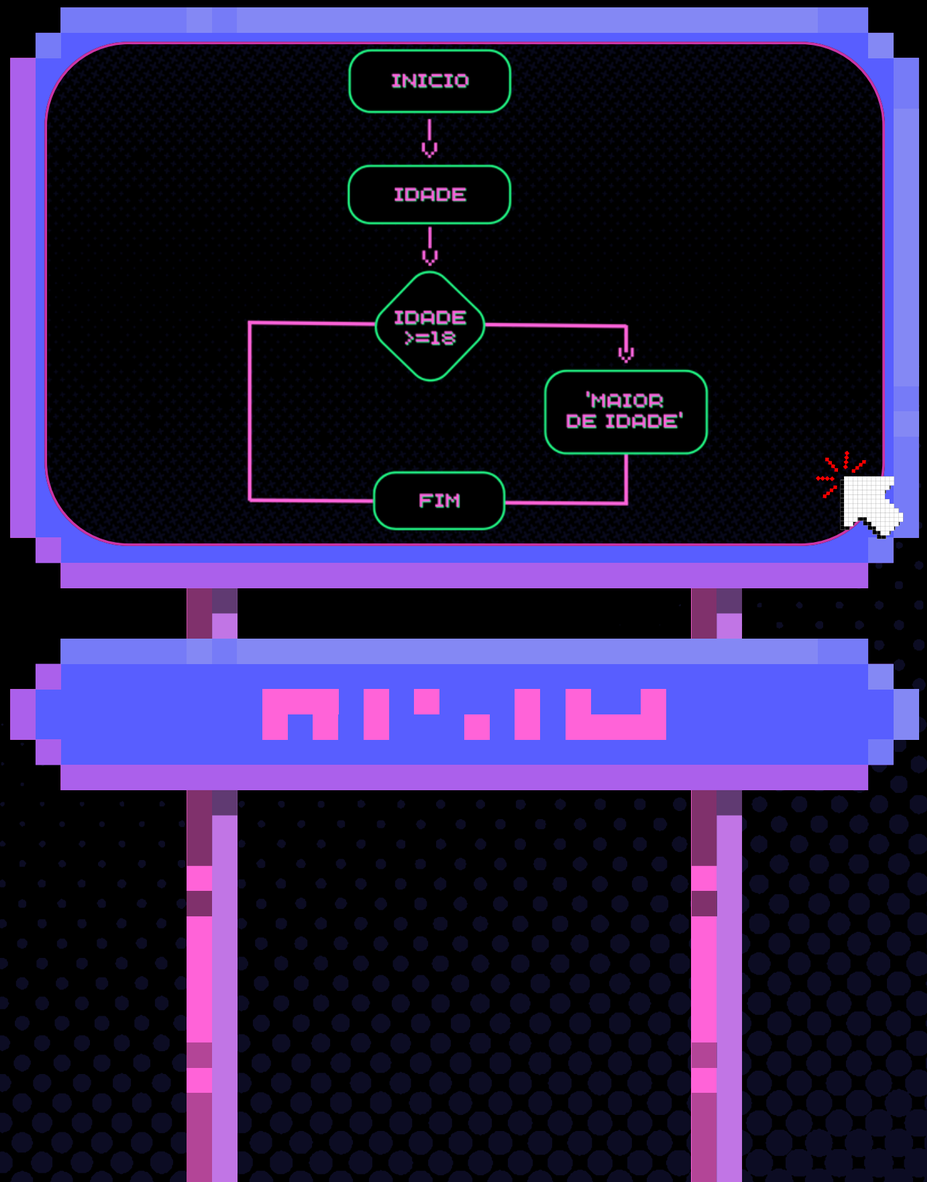
OPERADORES DE ASSOCIAÇÃO

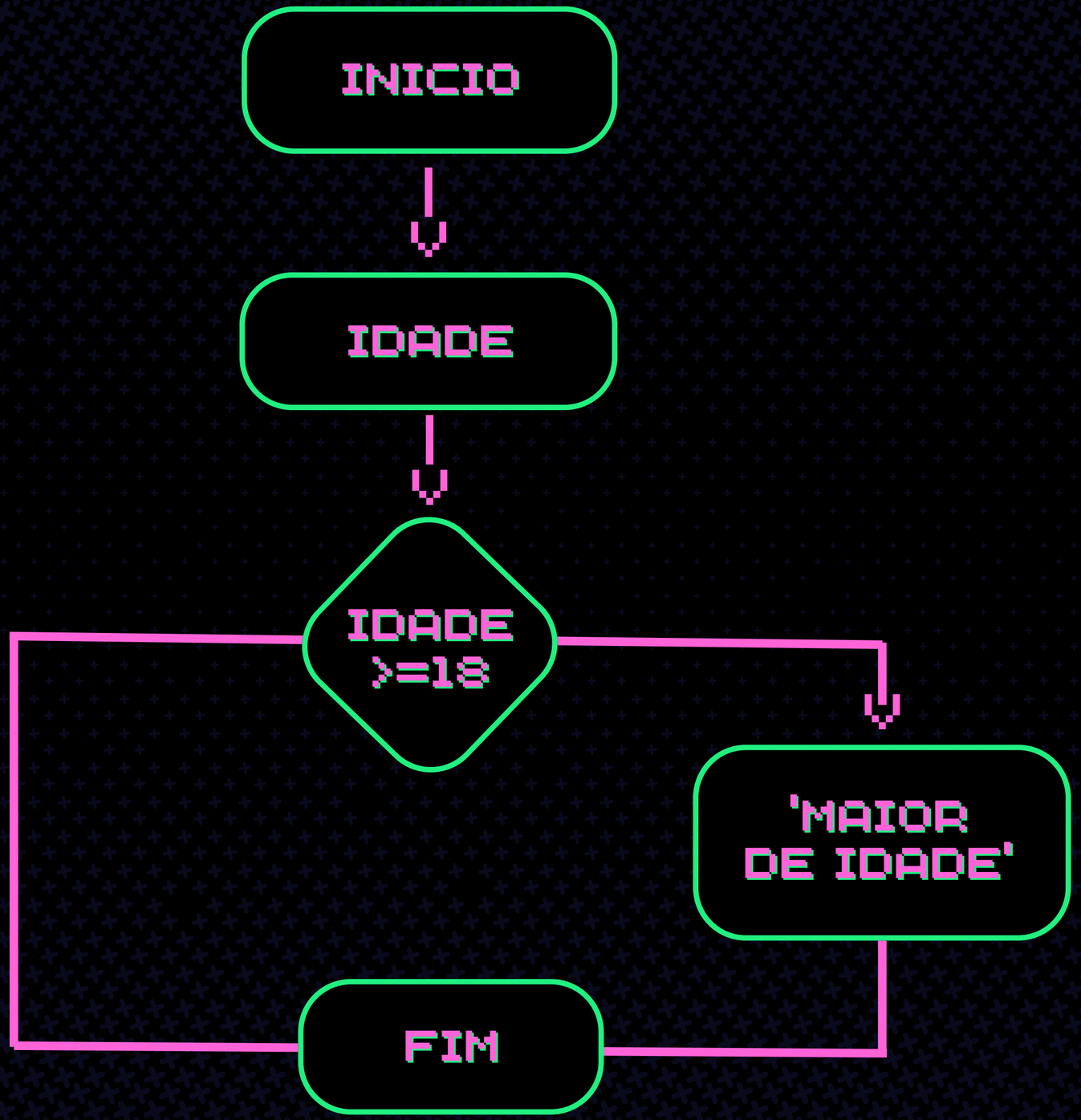
operador	conceito	**exemplo**
IN	retorna True caso o valor seja encontrado na sequência	2 in x
NOT IN	retorna True caso o valor não seja encontrado na sequência	2 not in x

MENU 01 07 12

OPERADORES CONDICIONAIS

EM PYTHON, UM PROGRAMA DEVE SER CAPAZ DE TOMAR DECISÕES COM BASE EM VALORES E RESULTADOS GERADOS DURANTE SUA EXECUÇÃO, OU SEJA, DEVE SER CAPAZ DE DECIDIR SE DETERMINADA INSTRUÇÃO DEVE OU NÃO SER EXECUTADA DE ACORDO COM UMA CONDIÇÃO.




INICIO
IDADE
IDADE
>=18
'MAIOR
DE IDADE'
FIM

```
1 numero = 10
2 if numero > 5:
3     print('o valor é maior que 5')
```

Resultado

```
o valor é maior que 5
```

```
1  numero = 3
2  if numero > 5:
3      print('o valor é maior que 5')
4  else:
5      print('o valor é menor ou igual a 5')
```

Resultado

```
o valor é menor ou igual a 5
```

```python
linguagem = 'python'
if linguagem == 'python':
    print(f'{linguagem} é uma linguagem de programação de alto nível')
elif linguagem == 'c++':
    print(f'{linguagem} é uma linguagem de programação compilada')
```

Resultado

```
python é uma linguagem de programação de alto nível
```

PLAYER 1

HIGHSCORE 2500

PLAYER 2

LAÇOS DE REPETIÇÃO

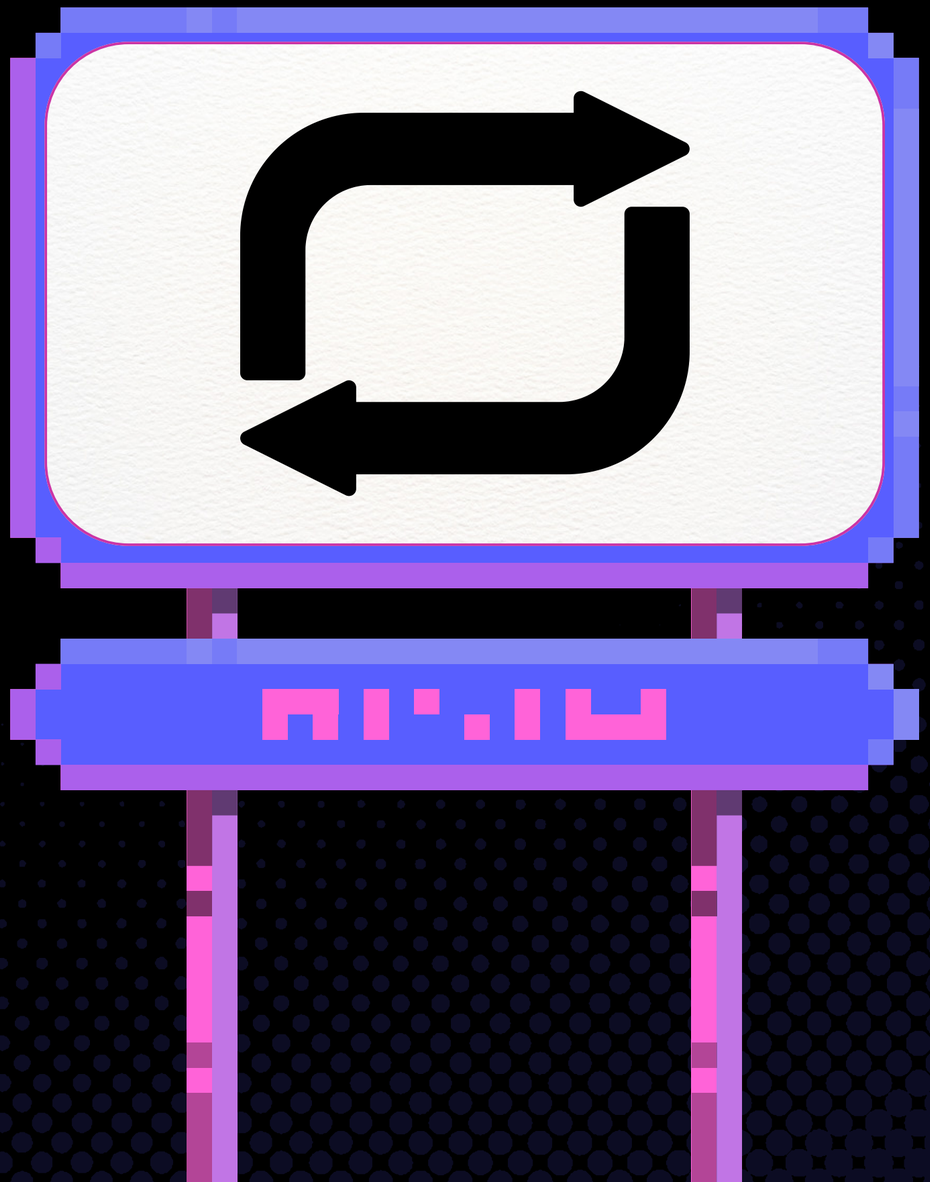
START MENU SIGN IN

MENU 01 07 12

RESUMO

LAÇOS DE REPETIÇÃO SÃO RECURSOS DAS LINGUAGENS DE PROGRAMAÇÃO RESPONSÁVEIS POR EXECUTAR COMANDOS REPETIDAS VEZES ATRAVÉS DE DETERMINADAS CONDIÇÕES ESPECÍFICAS.

O PYTHON TEM DUAS ESTRUTURAS DE REPETIÇÃO: FOR E WHILE.

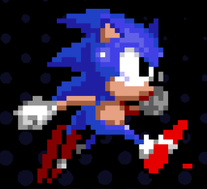


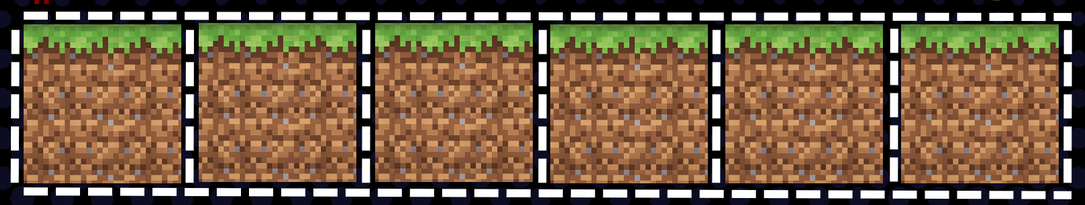
MENU

01

07

12



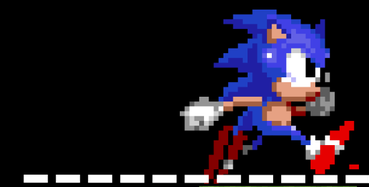


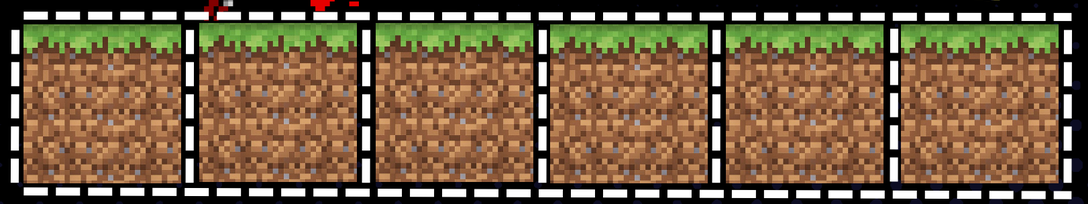
MENU

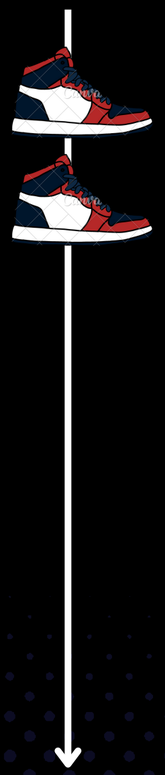
01

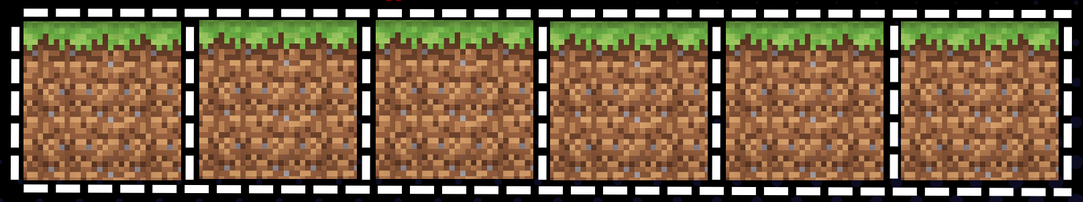
07

12







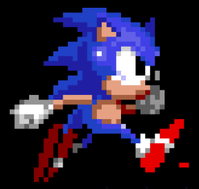


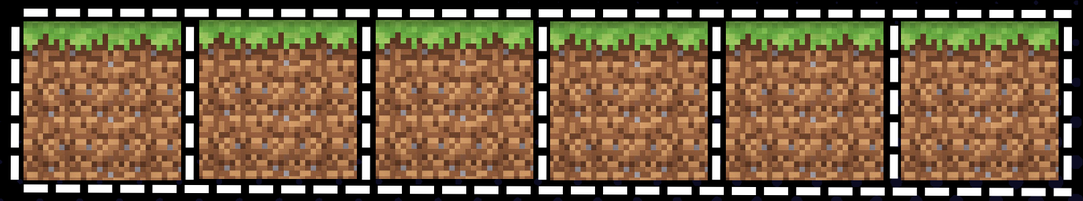
MENU

01

07

12



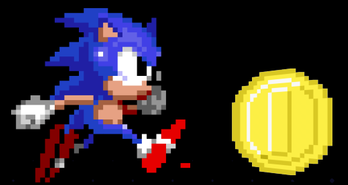


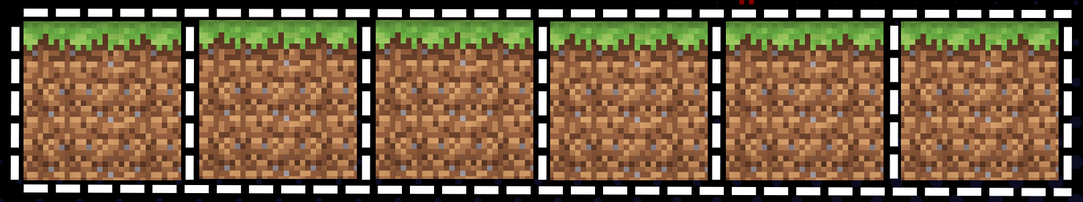
MENU

01

07

12



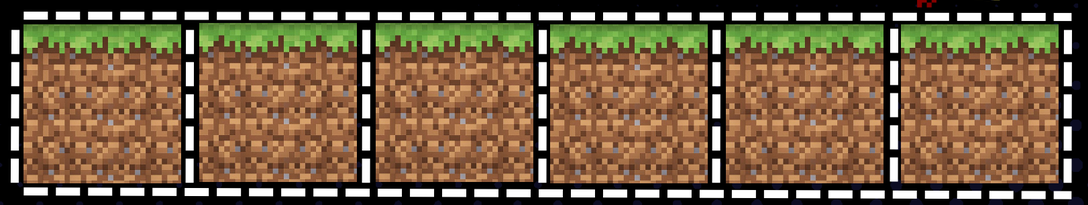


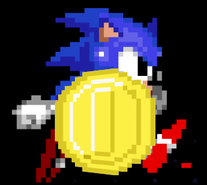
MENU

01

07

12



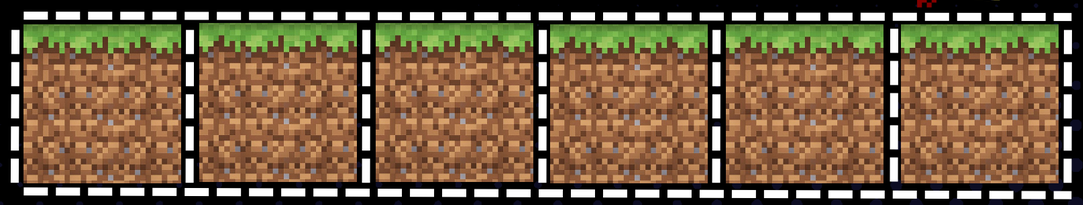


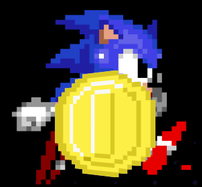
MENU

01

07

12



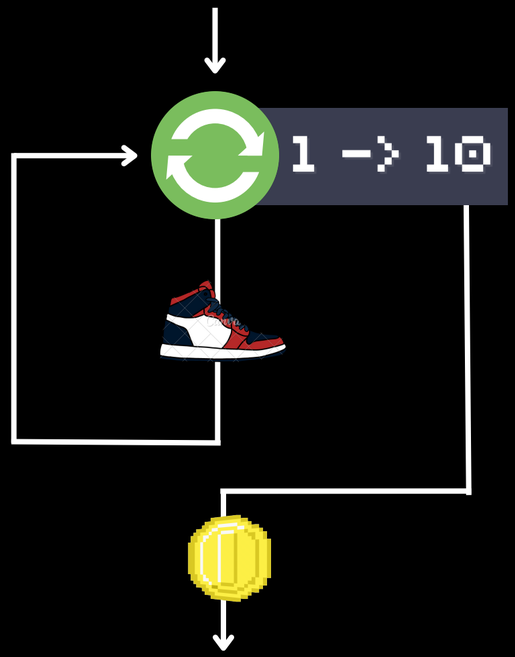


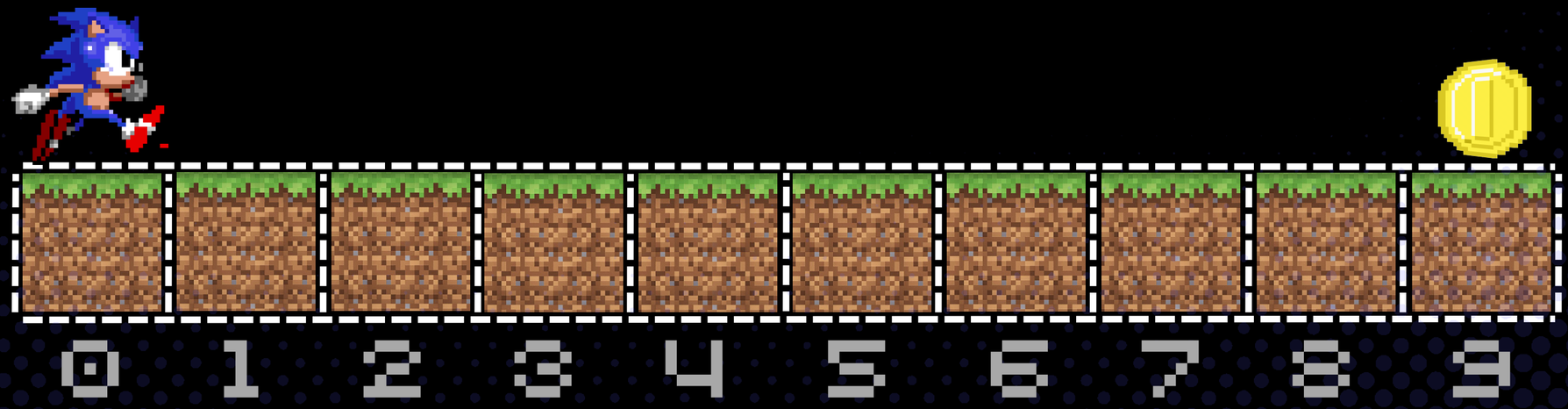
MENU

01

07

12





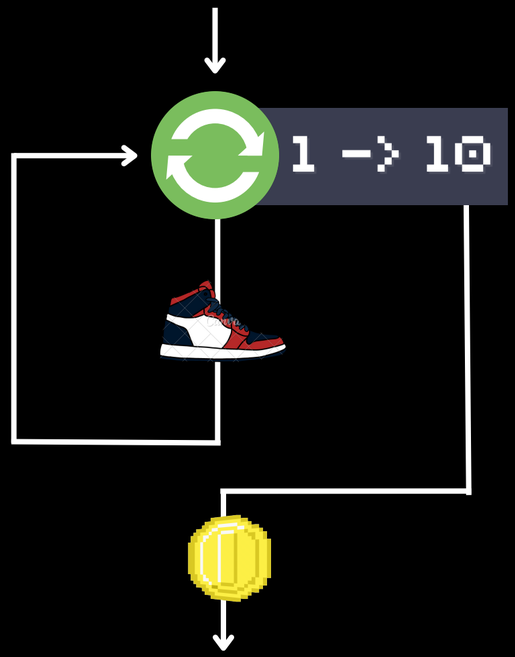
0 1 2 3 4 5 6 7 8 9

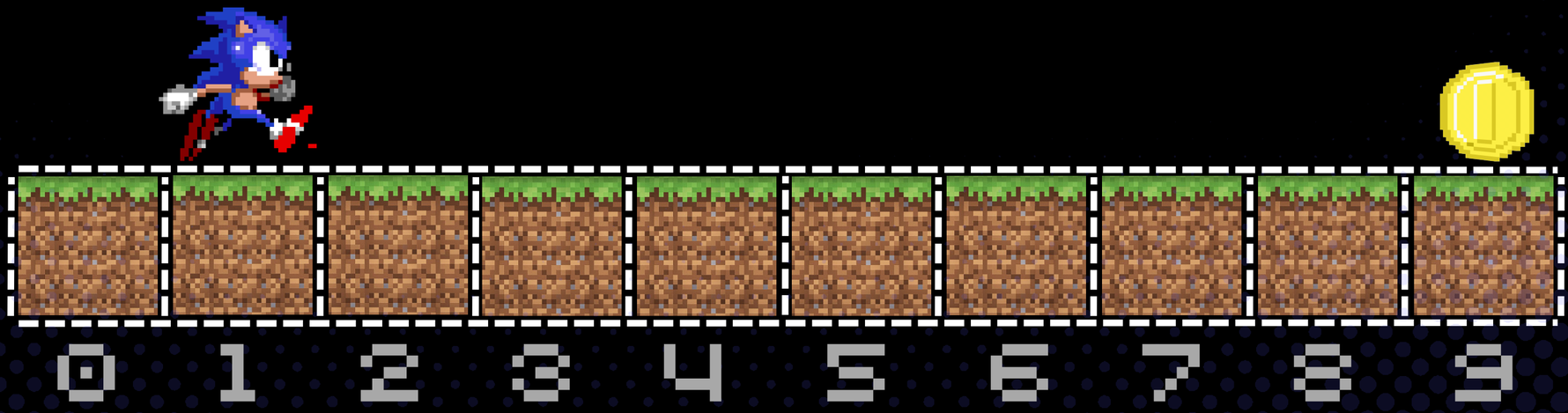
MENU

01

07

12





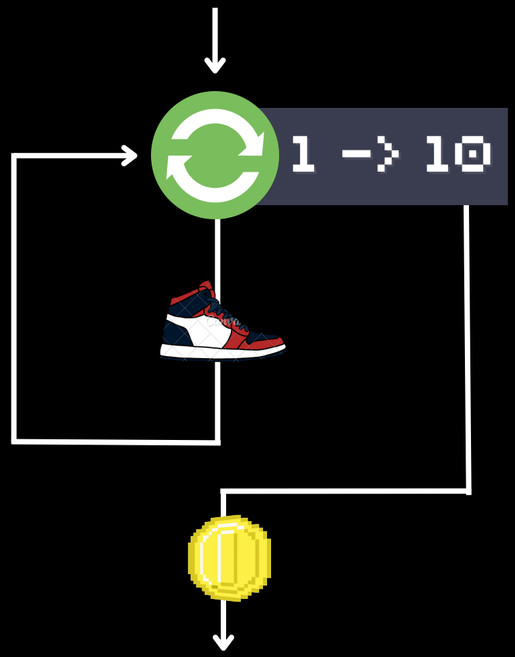
0 1 2 3 4 5 6 7 8 9

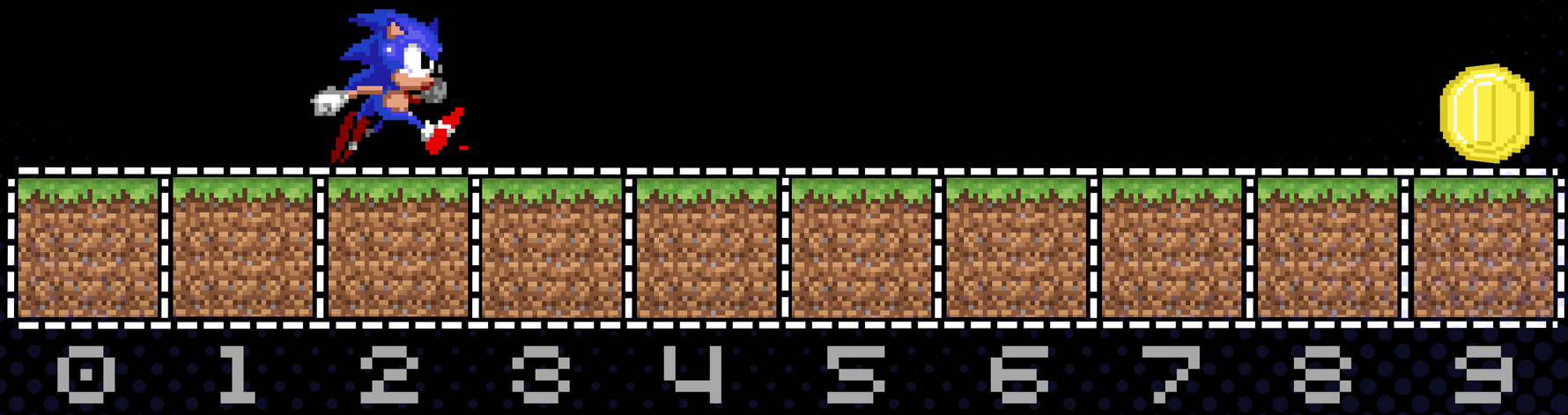
MENU

01

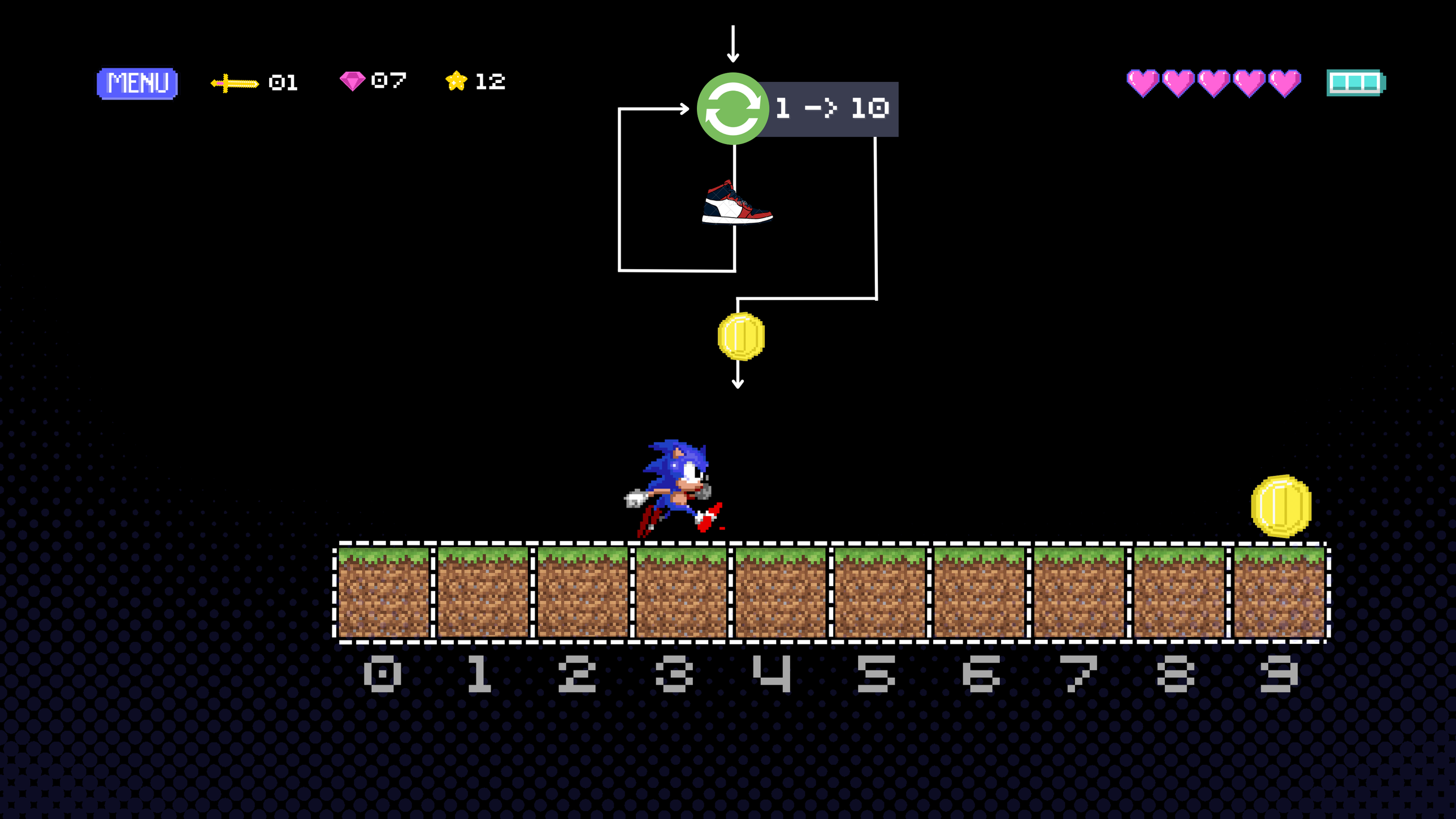
07

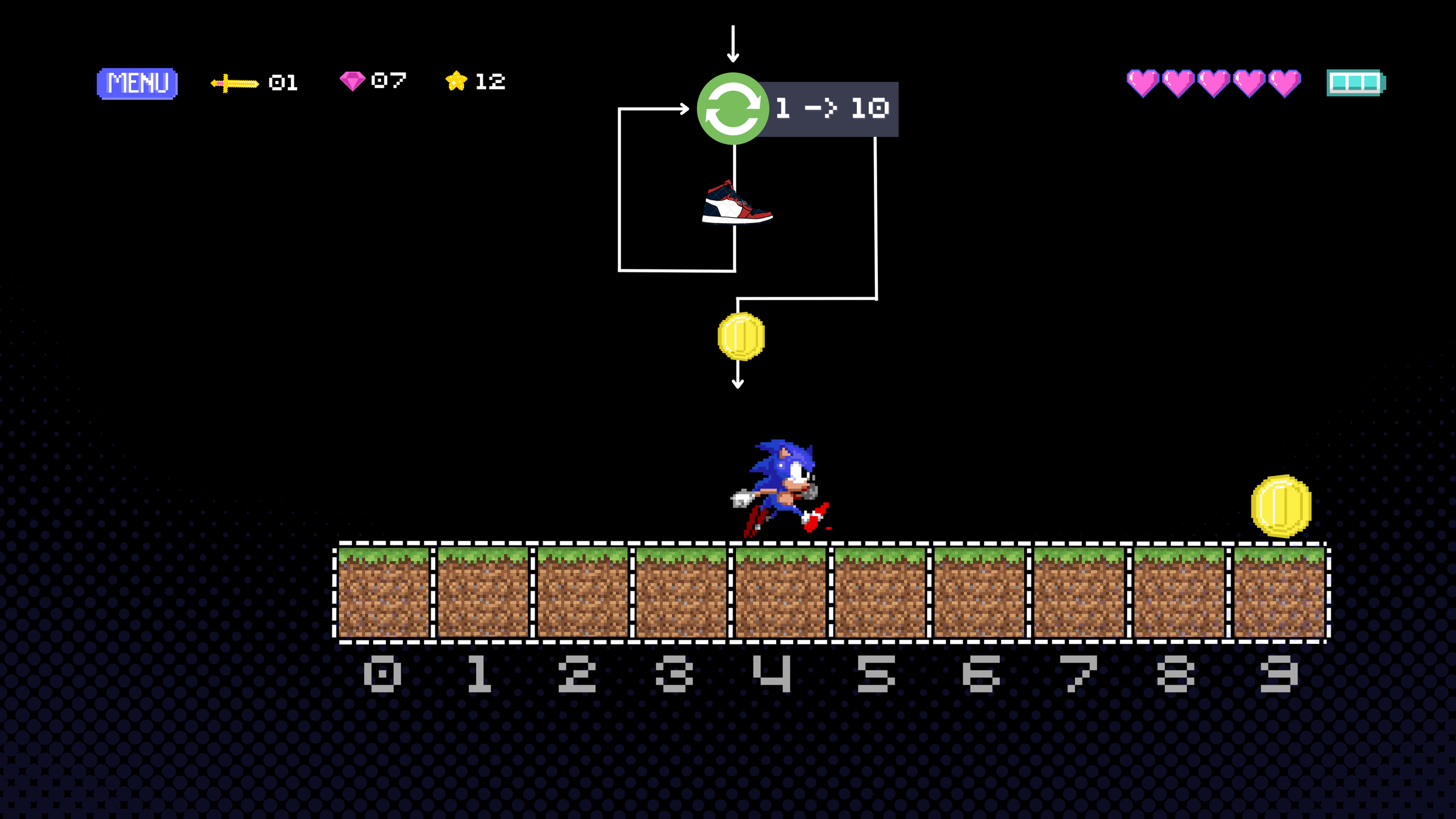
12

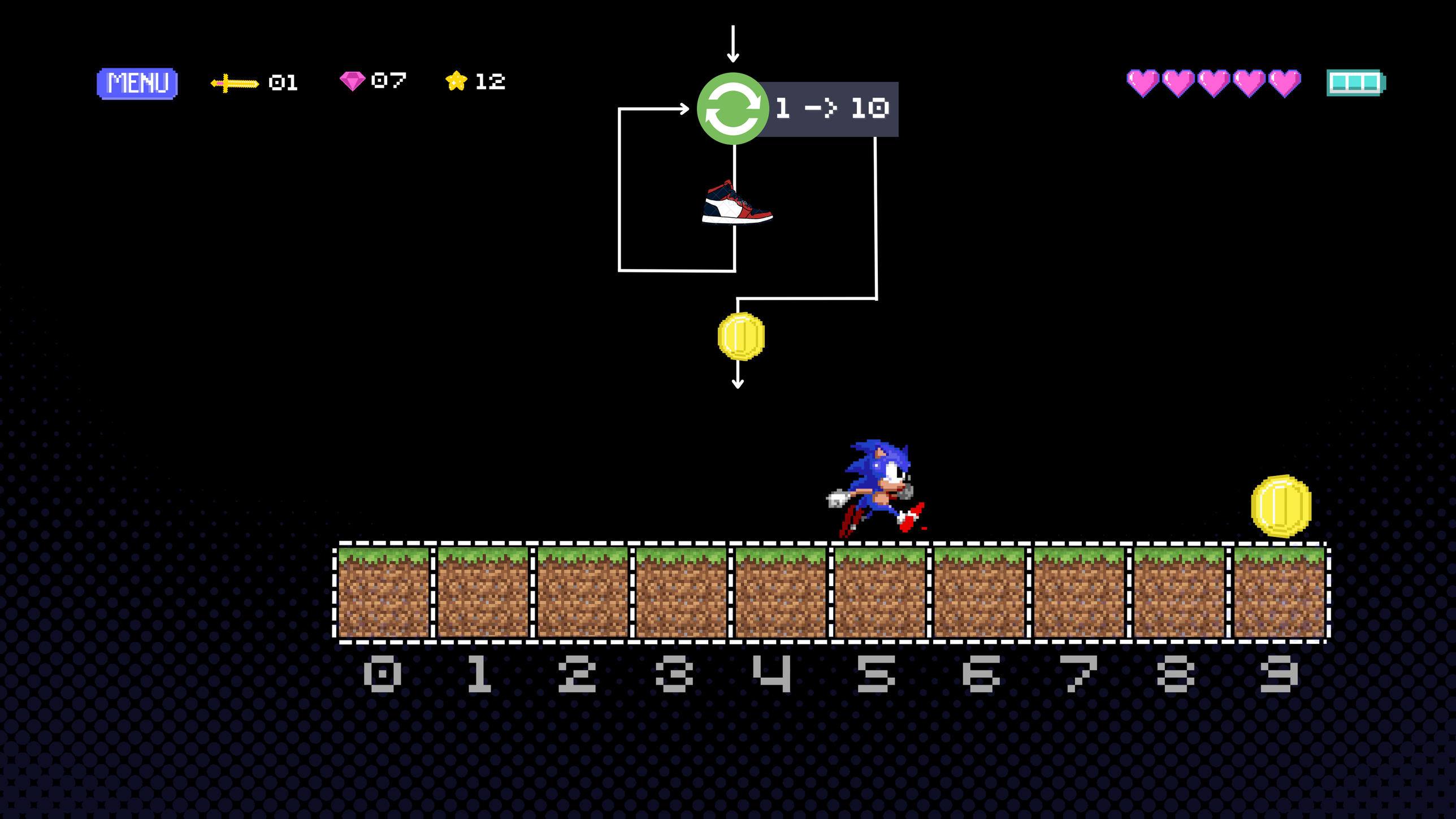


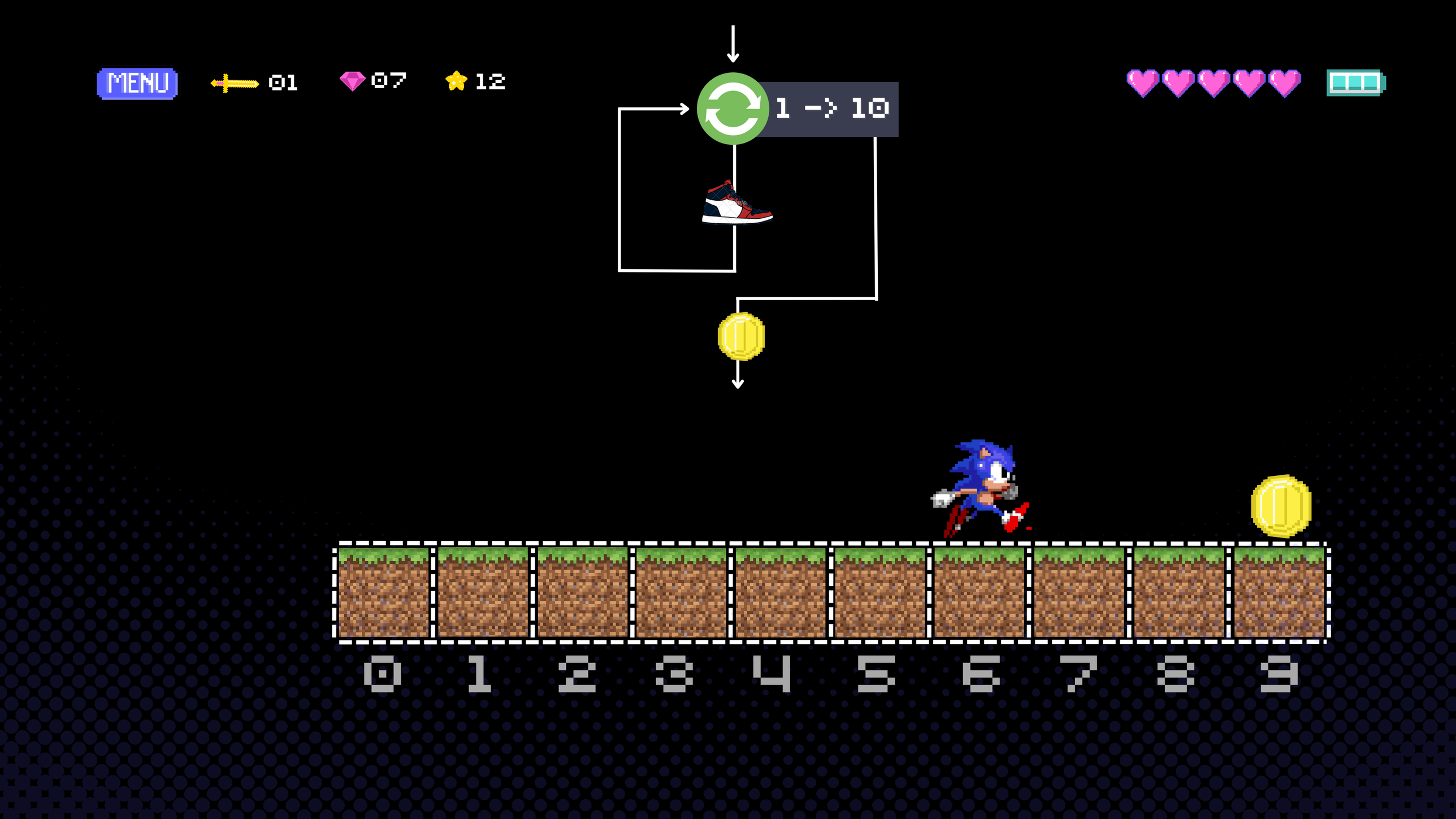


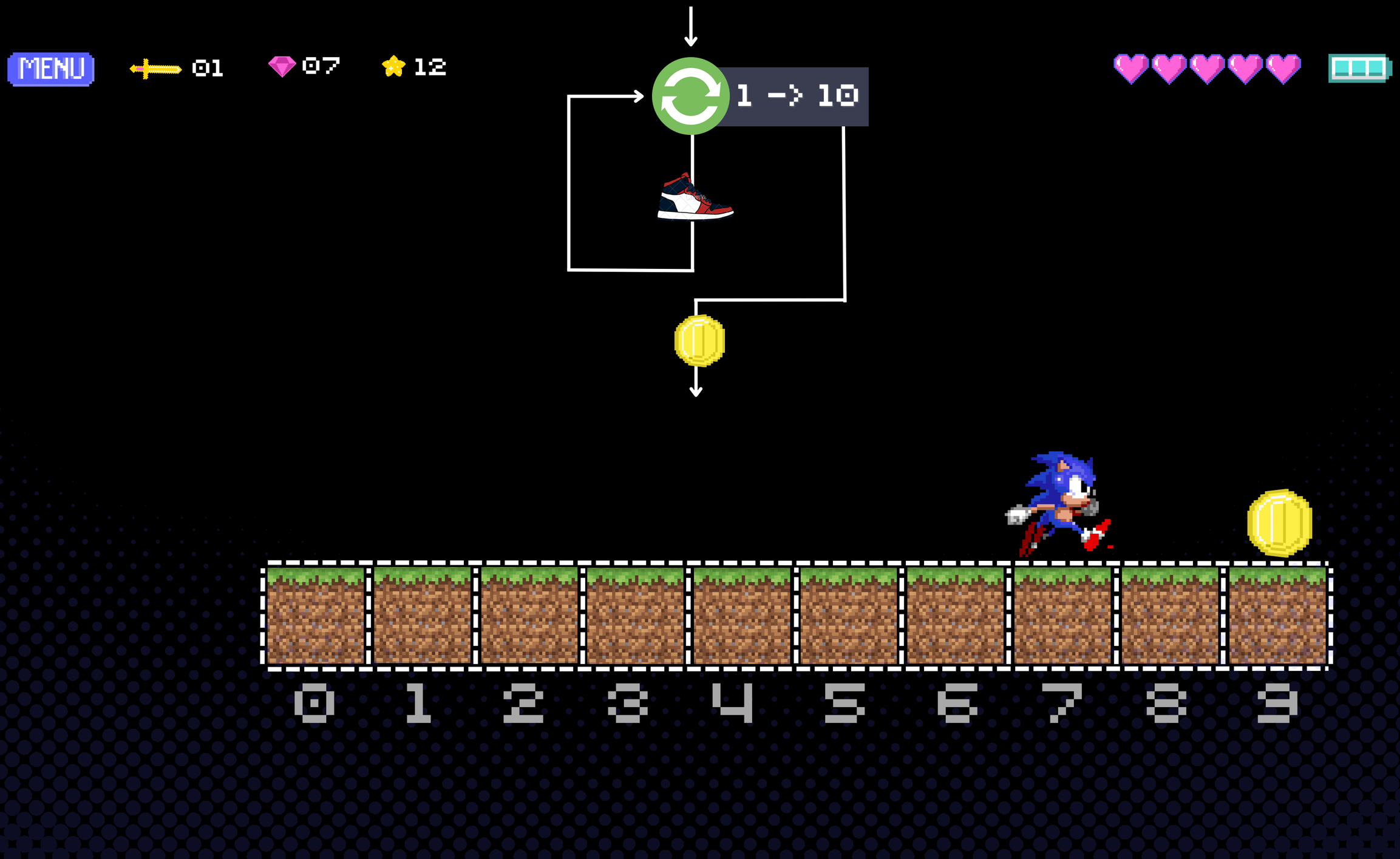
0 1 2 3 4 5 6 7 8 9


MENU
01
07
12
1 -> 10
0 1 2 3 4 5 6 7 8 9


MENU
01
07
12
1 -> 10
0 1 2 3 4 5 6 7 8 9


MENU
01
07
12
1 -> 10
0 1 2 3 4 5 6 7 8 9


MENU
01
07
12
1 -> 10
0 1 2 3 4 5 6 7 8 9

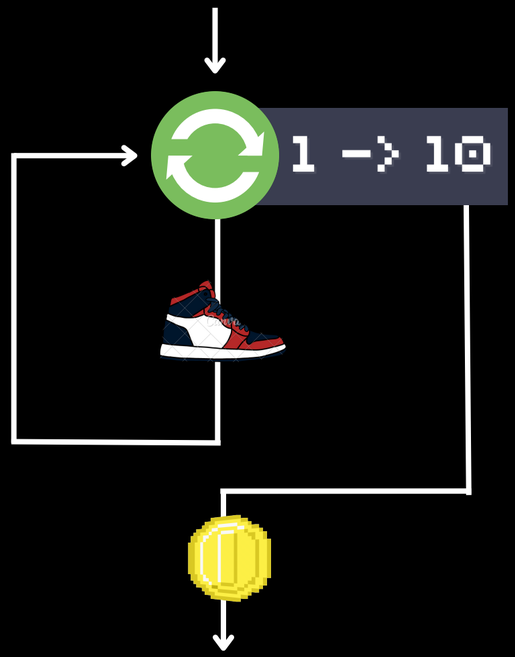

MENU
01
07
12
1 -> 10
0 1 2 3 4 5 6 7 8 9

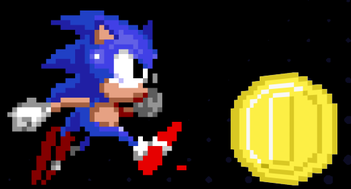
MENU

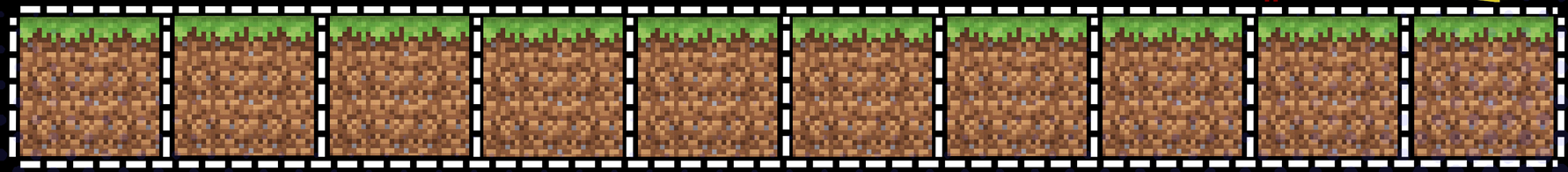
01

07

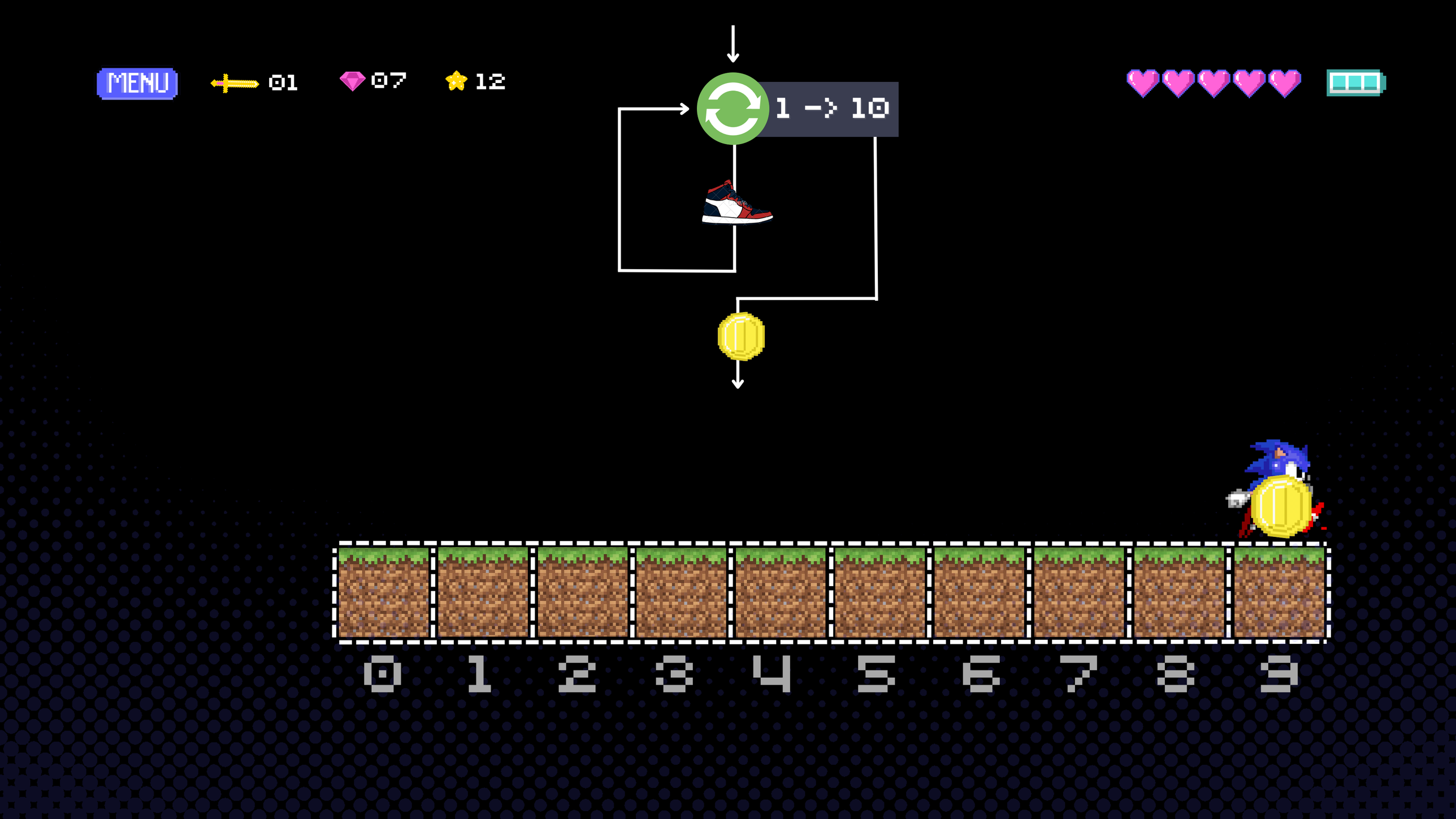
12



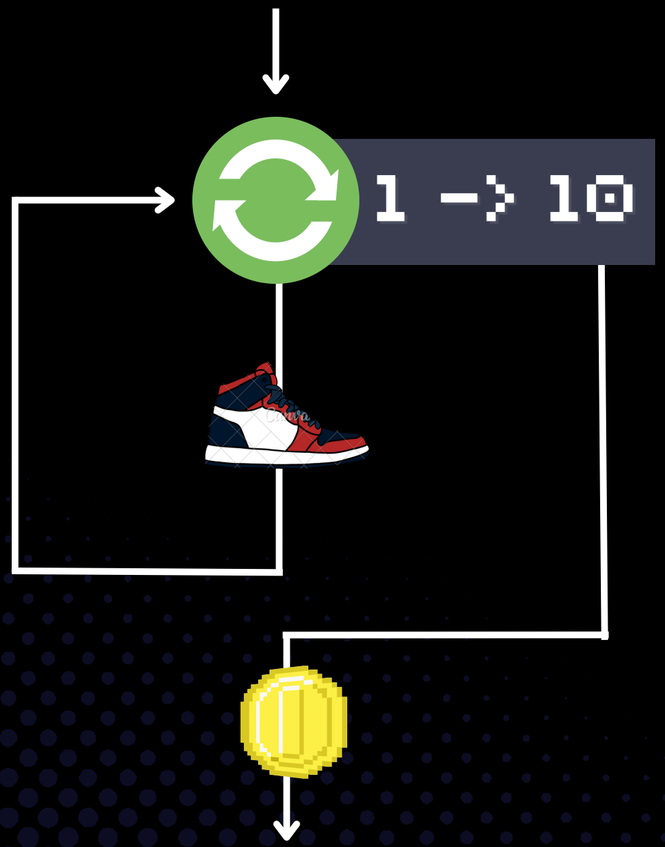




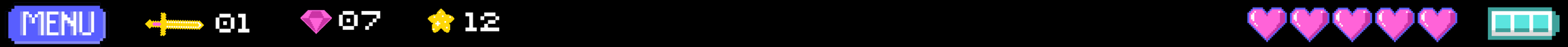
0 1 2 3 4 5 6 7 8 9


MENU
01
07
12
1 -> 10
0 1 2 3 4 5 6 7 8 9

MENU 01 07 12




```
laço i no intervalo(1, 10)
    passo
pega
```




```
laço i no intervalo(1, 10)
    passo
pega
```

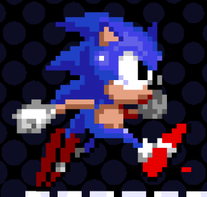
```
for i in range(1, 10):
    passo
pega
```

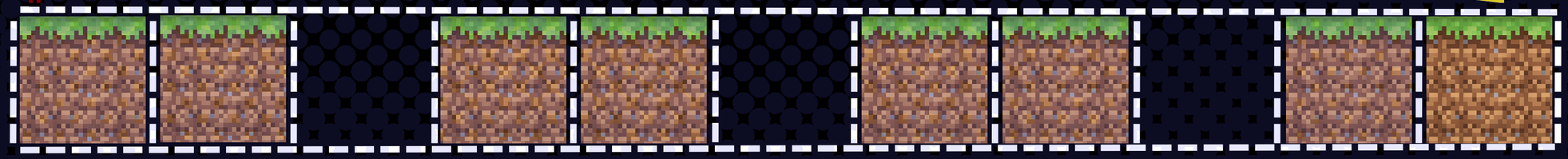
MENU

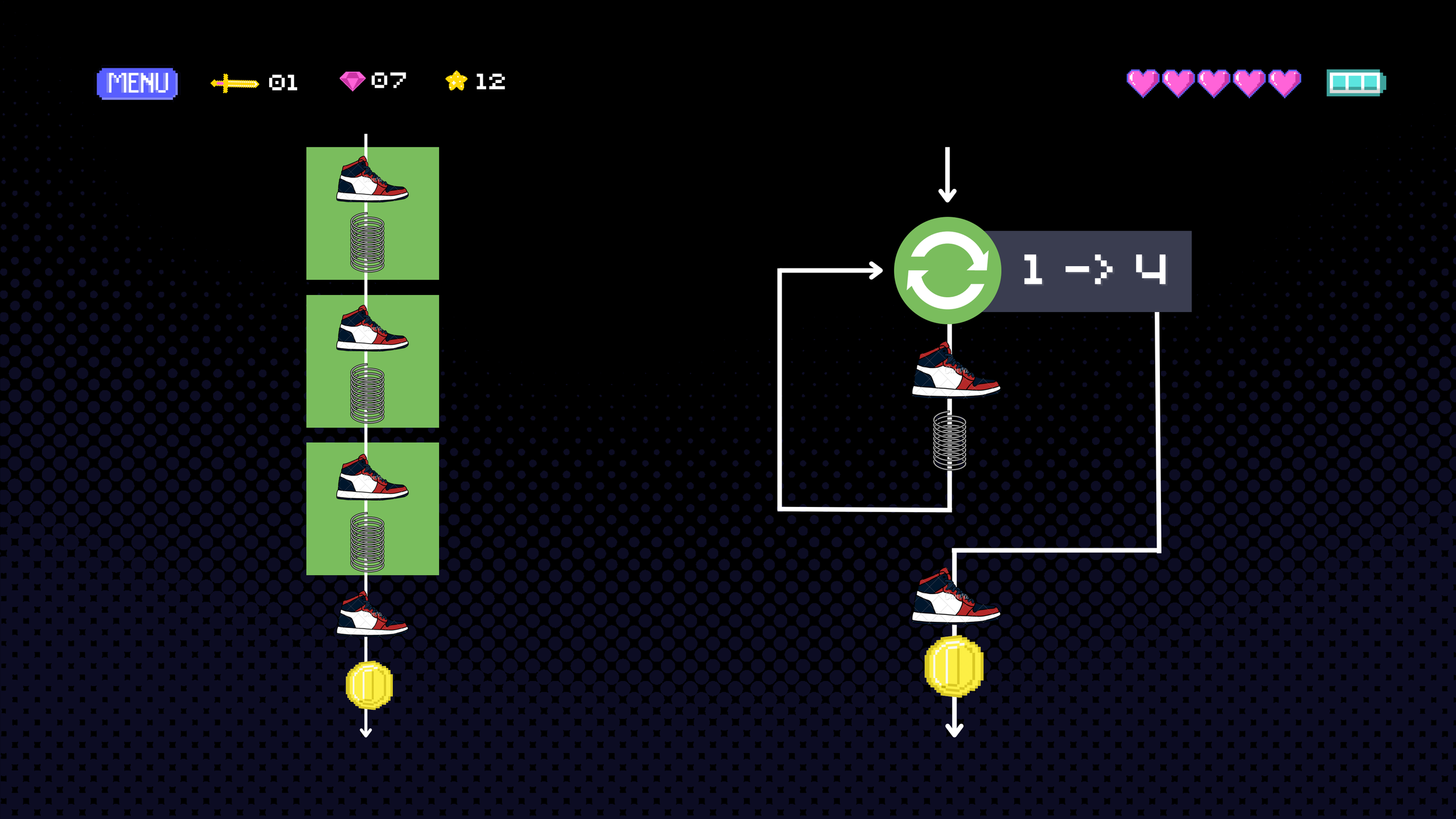
01

07

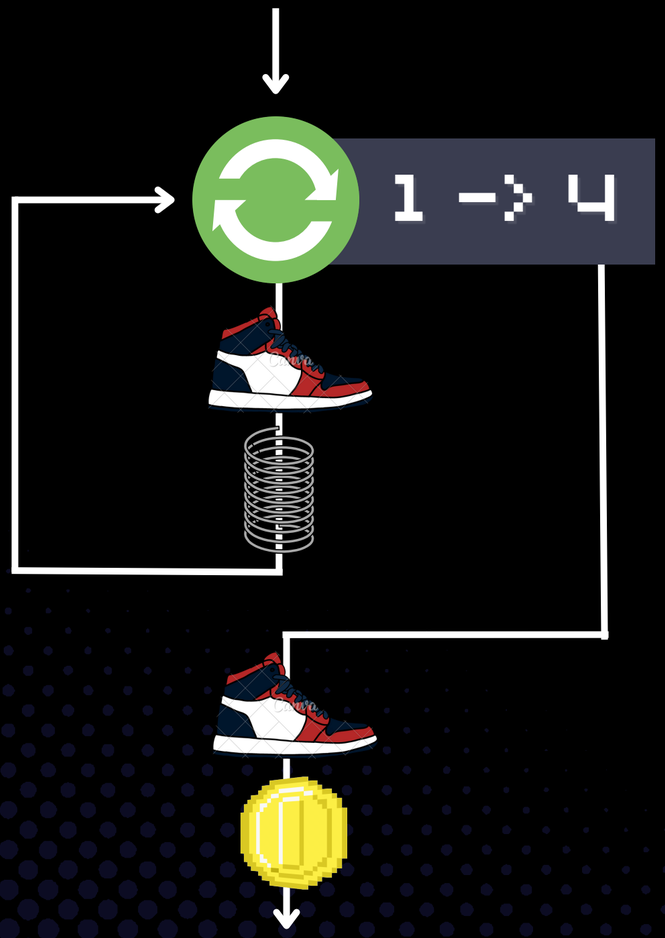
12





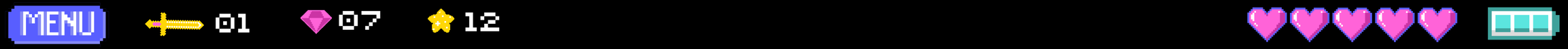

MENU
01
07
12
1 -> 4

MENU 01 07 12




```
laço i no intervalo(1, 4)
    passo
    pula
passo
pega

passo
pula
passo
pula
passo
pula
passo
pega
```




```
laço i no intervalo(1, 4)
    passo
    pula
passo
pega
```

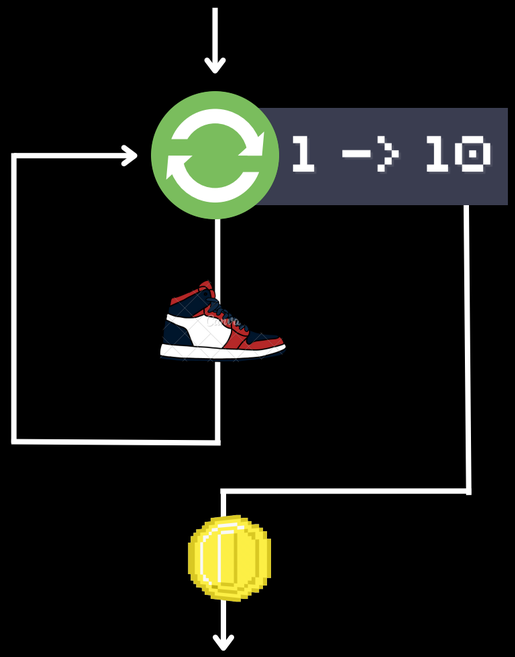
```
for i in range(1, 4):
    passo
    pula
passo
pega
```

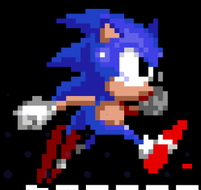
MENU

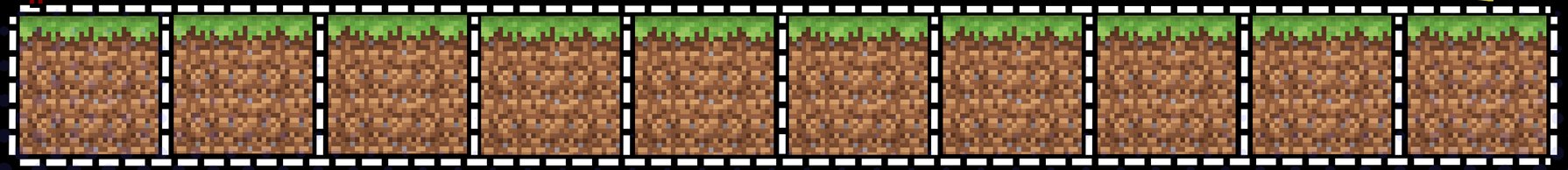
01

07

12







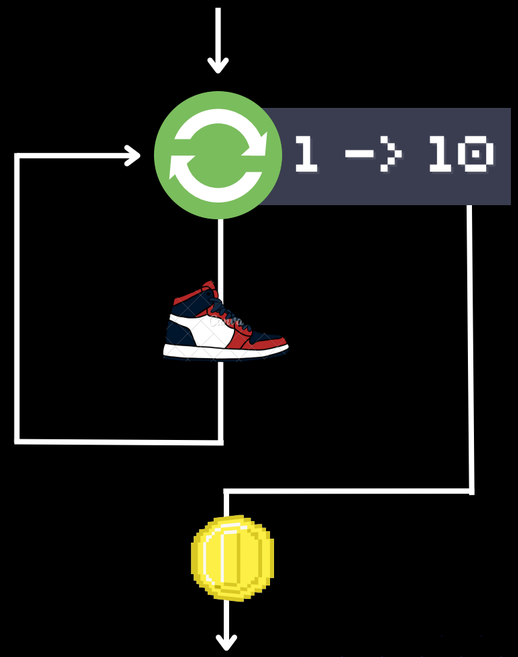
0 1 2 3 4 5 6 7 8 9

MENU

01

07

12



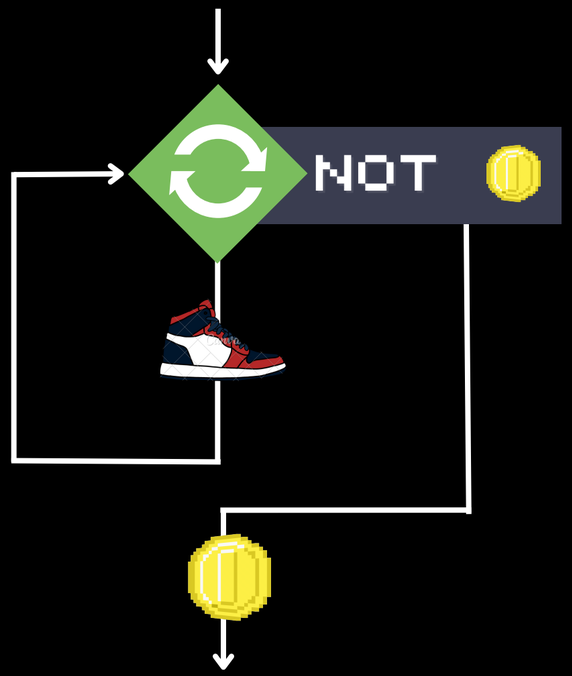
0 1 2 3 4 5 6 7 ?

MENU

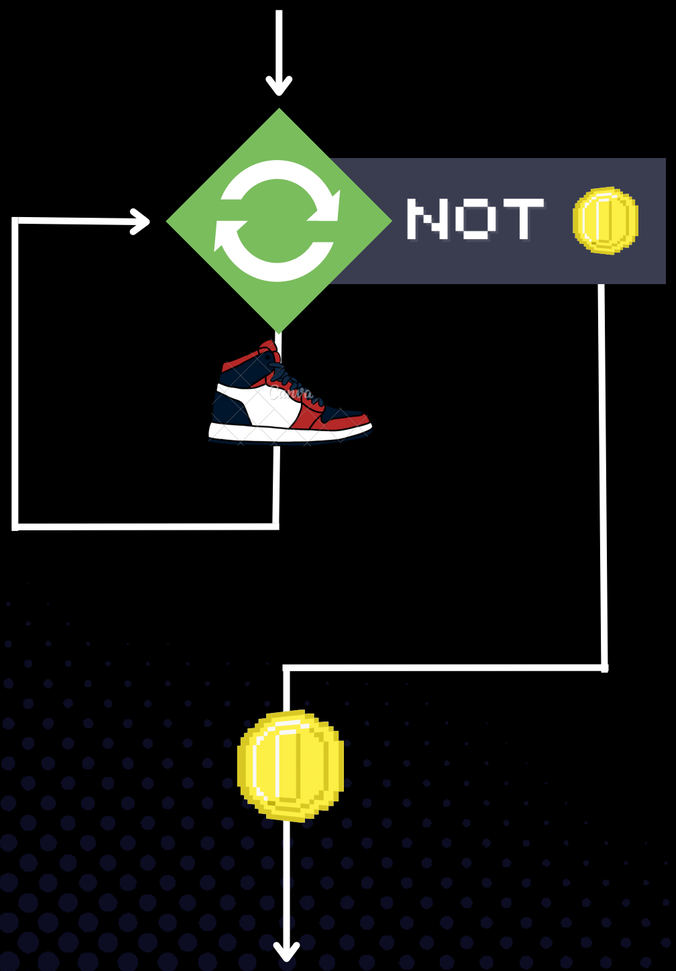
01

07

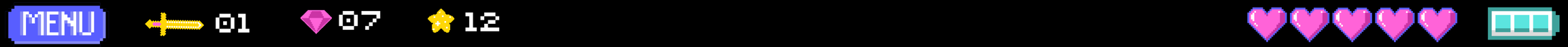
12



0 1 2 3 4 5 6 7 ?




```
enquanto não
        passo
pega
```

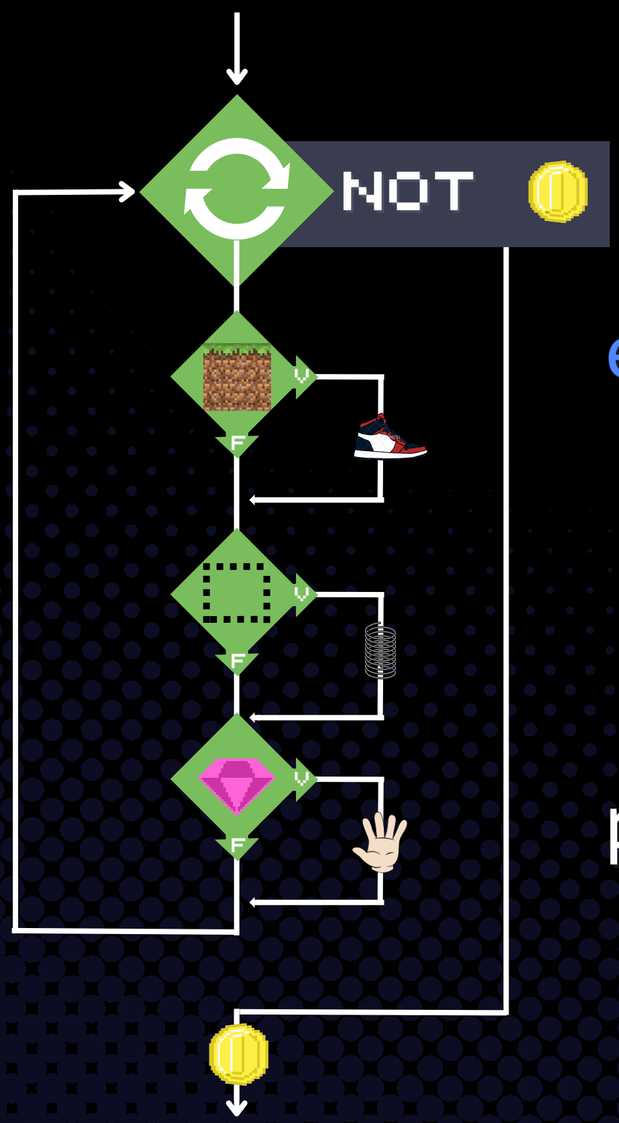



```
enquanto não 🪙
    passo
pega
```

```
while not 🪙:
    passo
pega
```


MENU
01
07
12
NOT
V
F
V
F
V
F

MENU 01 07 12

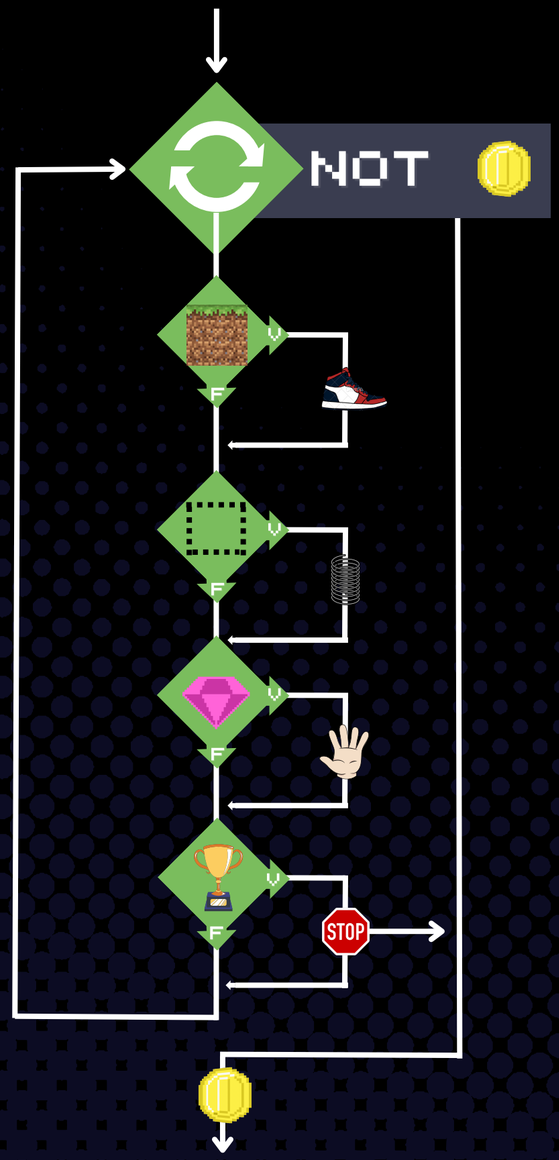



```
enquanto não
    se
        passo
    se
        pula
    se
        pega
pega
```

```
while not :
    if :
        passo
    if :
        pula
    if :
        pega
pega
```


MENU
01
07
12
NOT
V
F
V
F
V
F
V
F
STOP

MENU 01 07 12




```
enquanto não
    se
        passo
    se
        pula
    se
        pega
    se
        interrompa
pega
```

```
while not :
    if :
        passo
    if :
        pula
    if :
        pega
    if :
        break
pega
```