

Linguagem Musical


Uma linguagem de programação com notações musicais



Notações Utilizadas:

Notas e Pausas

Notações utilizadas

	colcheia	if
	semicolcheia	else
	Pausa de mínima	Fim de linha
	Pausa de colcheia	comentário
	Pauta com clave de sol	input





Acidentes e intervalos

Notações utilizadas

8^{va}	Oitava à cima	multiplicação
8^{vb}	Oitava abaixo	divisão
b	bemol	subtração
\sharp	sustenido	soma
\natural	bequadro	igual


Repetições e codas

Notações utilizadas

	Repetição de compasso	igual comparativo
	coda	return
	segno	Definição de função
<i>D.S.</i>	Dal segno	Chamada de função
	Barras de repetição	while

Dinâmica e Ornamentos

Notações utilizadas

<i>tr</i>	trinado	ou
≈	gruppetto	true
∞	Grupetto invertido	false
~	mordente	not
	Crescendo e diminuendo	Menor e maior

Outros

Notações utilizadas

	cluster	and
	microfone	print
,	respiração	Vírgula para separação de parâmetros
	Blocos de início e fim	Início e fim de bloco (funções)

Relação entre as notações e suas funções



Aritméticas

SOMA:

- O sustenido acrescenta na frequência da nota o equivalente a 1 semitom.

SUBTRAÇÃO

- O bemol subtrai da frequência da nota um semitom.

IGUAL:

- O bequadro é usado para retirar um acidente. A nota tem seu valor “natural”.

MULTIPLICAÇÃO:

- A oitava acima tem o dobro da frequência da fundamental.

DIVISÃO:

- A oitava abaixo tem metade frequência da fundamental.

Lógicas e Comparativas

AND:

- O cluster é a união de muitas notas que não pertencem à mesma tonalidade.

OR:

- O trilo diz que nota deve oscilar do início ao fim entre ela mesma e seu semitom acima.

NOT:

- O mordente diz que nota deve uma vez, rapidamente subir pra o semitom a cima e voltar.

DIMINUENDO:

- O diminuendo diz que a dinâmica daquele pedaço da música deve diminuir gradativamente.

CRESCENDO:

- O crescendo diz que a dinâmica daquele pedaço da música deve aumentar gradativamente.

IGUAL COMPARATIVO:

- A repetição de compasso diz que o próximo compasso deve ser igual ao que está sendo tocado no momento.

Funções, while e if

DEFINIÇÃO DE FUNÇÃO:

- O segno marca um pedaço da música que deve ser tocado sempre que necessário.

CHAMADA DE FUNÇÃO:

- O dal segno indica que o segno deve ser tocado.

RETURN:

- A coda costuma ser a parte final da música e é chamada quando todas as outras partes já foram tocadas.

IF:

- A colcheia é uma nota que dura um oitavo do “tempo”

ELSE:

- A semicolcheia é uma nota que dura 1/16 do “tempo”

WHILE:

- As barras definem o início e o fim do bloco que deve ser repetido.

Outros

PRINT:

- O microfone capta o som e, se ligado a um amplificador “tira a música da partitura”

INPUT:

- Na pauta a música deixa de ser passada oralmente e é registrada. Sai do interlocutor e se transforma em código.

VÍRGULA:

- A respiração é marcada para que o músico dê espaço entre as notas tocadas. Ela dá naturalidade para as frases musicais.

FIM E INÍCIO DE BLOCO DE FUNÇÃO:

- As barras delimitam o fim e o início da música. Nesse caso seria de cada parte da música. Geralmente as barras duplas simples seriam utilizadas, mas nesse caso pareceu fazer mais sentido enfatizar o início e o fim.

FINAL DE LINHA:

- A pausa de mínima dura dois tempos.

COMENTÁRIOS:

- A pausa de colcheia dura $\frac{1}{8}$ de tempo.

EBNF

PROGRAM = { COMMAND } ;

COMMAND = ((λ | ASSIGNMENT | PRINT | FUNCCALL | RETURN), " ") | WHILE | IF | FUNCASSIG | COMMENT ;

ASSIGNMENT = IDENTIFIER, "←", RELEXP ;

PRINT = "🖨️", EXPRESSION ;

WHILE = "!", "(", RELEXP, ")", CODE, "!!";

IF = ("♪", "(", RELEXP, ")", BLOCK, "♪") | ("♪", "(", COND, ")", BLOCK, "♪", " ", CODE, "♪");

FUNCASSIG = "⌘", "!", VAR, "(", PARAMASSIG, ")", BLOCK, "!!";

PARAMCALL = [RELEXP { " " RELEXP }] ;

PARAMASSIG = [IDENTIFIER { " " IDENTIFIER }] ;

RETURN = "🏠", [RELEXP], " " " ;

RELEXP = EXPRESSION , { ("⌘" | ">" | "<"), EXPRESSION } ;

EXPRESSION = TERM, { ("#" | "↳" | "⌘"), TERM } ;

TERM = FACTOR, { ("⌘" | "⌘" | "⌘"), FACTOR } ;

FACTOR = ("#" | "↳" | "⌘"), FACTOR) | NUMBER | (⌘ IDENTIFIER "(" PARAMCALL ")") | "(", RELEXP, ")" | IDENTIFIER | INPUT ;

INPUT = "🎵" ;

EBNF

IDENTIFIER = LETTER, { LETTER | DIGIT | "_" };

NUMBER = DIGIT, { DIGIT };

LETTER = (a | ... | z | A | ... | Z);

DIGIT = (1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0);

COMMENT = " , "{*};

Exemplos



If e else

♪(4 5)

início do if

🎤 (10) _

print dentro do if

♪

fim do if

♪

início do else

🎤 (0) _

print dentro do

else

♪

fim do else

—

while

i = 5
variável

atribuição de

(i > 0)

início do while(cond)

while
print(i)

print dentro do

i = i - 1

diminua 1 de i

:

fim do while

Print, input e comentário

v h 

input para variável

 (v).

print da variável

, comentário,

comentário

função

⌘ fun(d,r)⌋

declara função fun

🗨 (d)⌋

print dentro da fun

⌈ r ⌋

retorno da fun

⌋

fim da fun

m ⌈ 1 ⌋
variável

atribuição de

g ⌈ _{ds} fun(m,7)⌋

variável igual à fun

🗨 (g)⌋
fun—

print do retorno da