

Linguagem Musical

Uma linguagem de programação com notações musicais



Notações Utilizadas:

Notas e Pausas

Notações utilizadas

 - colcheia

 - semicolcheia

 - Pausa de mínima

 - Pausa de colcheia

 - pauta com clave de sol

—

Acidentes e intervalos

Notações utilizadas

8^{va} - oitava acima

8^{vb} - oitava abaixo

b - bemol

\sharp - sustenido

\natural - bequadro

—

Repetições e codas

Notações utilizadas

∴ - repetição de compasso

⦿ - coda

‰ - segno

||:| - barras de início e fim de repetição

D.S. - dal segno

—

Dinâmica e Ornamentos

Notações utilizadas

tr - trilo

∞ - grupetto

∞ - grupetto invertido

~ - mordente

< > - crescendo e diminuendo

—

Outros

Notações utilizadas



- cluster



- microfone



- respiração



- blocos de início e fim



Relação entre as notações e suas funções



Aritméticas

A soma é representada pelo sustenido.

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

A subtração é representada pelo bemol.

- O bemol subtrai da frequência da nota um semitom.

O igual é representado pelo bequadro

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

A multiplicação é representada pelo intervalo de oitava acima.

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

A divisão é representada pelo intervalo de oitava abaixo.

- A oitava abaixo tem metade frequência da fundamental.

Lógicas e Comparativas

O and é representado pelo cluster.

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

O or é representado pelo trilo.

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

O not é representado pelo mordente.

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

O maior é representado pelo diminuendo.

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

O menor é representado pelo crescendo.

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

O igual comparativo é representado pela repetição de compasso.

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

Funções, while e if

A definição de função é representada pelo segno.

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

A chamada de função é representada pelo dal segno.

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

O return é representado pela coda.

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

O if é representado pela colcheia.

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

O else é representado pela semicolcheia.

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

O while é representado pelas barras de início e fim da repetição

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

Outros

O print é definido pelo microfone.

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

O input é a pauta com clave de sol.

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

A vírgula é representada pela respiração.

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

Os blocos de cada função são representados pelas barras de início e fim.

- 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.

O final de linha é representado pela pausa de mínima.

- A pausa de mínima dura dois tempos.

Os comentários são representados pela pausa de colcheia

- 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 

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 ⬅ _{D.S.} fun(m,7)⌘

variável igual à fun

🗨 (g)⌘
fun⌘

print do retorno da