Manual

João Gabriel Basi

8 de julho de 2016

1. Python

1.1. Indentação

- A indentação é feita por espaços, enter e dois pontos

1.2. Variáveis

- Os tipos das variáveis em Python são determinadas pelo compilador, mas podem ser declarados ao declarar a variável
- Strings são sempre consideradas como listas de caracteres

1.3. Listas

- Podem mesclar tipos de variáveis
- São declaradas como:
  - lista1 = []
  - lista2 = [0]*5 (para 5 posições com zero)
  - lista3 = [0 for i in range(5)] (mesmo do de cima)
  - lista4 = [0, 0, 0, 0, 0]
- Podem ser do tipo array, usando [] (os valores podem ser alterados à vontade), tupla, usando () (os valores não podem ser alterados) ou set, usando {} (não pode ter valores repetidos)
- São referenciados como:
  list[0] = 0

1.4. Funções

- Tipos dos argumentos não precisam ser fornecidos
- São declaradas como:

```
def area(altura, largura):
    return altura * largura
```

1.5. Objetos

1.6. Expressões

1.6.1. Condicionais

- if
- elif
- else

1.6.2. Loops

- while (normal)
- O for é escrito de outro jeito:
  for i in range($< inicio >, < fim + 1 >, < passo >$):
  (O que cada loop faz)

1.6.3. Outros

-

1.7. Comparadores e booleanos

- == e !=
- <= e >=
- < e >
- && ou and
- || ou or
- ! ou not

1.8. Operadores

- + e −
- * e /
- ** (potenciação)
- % (módulo)
- // (divisão inteira)

1.9. Bibliotecas

- São importadas de um dos modos:
  - import <nome>
  - import <nome> as <novo nome>
  - from <nome> import <função> (para importar uma ou mais funções sem precisar de referência)
  - from <nome> import * (para importar todas as funções sem precisar de referência)
- Mais usadas:
  - matplotlib.pyplot: gráficos
  - numpy ou math: matemática
  - random: função random

1.10. Input e output

- Input: input() (retorna uma string, então tem que ser transformado para o tipo desejado)
- Output: print()

1.11. Comentários

- Só podem ser feitos de linha em linha com #

1.12. Funções e métodos úteis

- len(list)
- Métodos das listas, tuplas e sets

2. C e C++

1.1. Indentação

- A indentação é feita por chaves e ponto e vírgula

1.2. Variáveis

- Os tipos das variáveis têm que ser especificados ao serem criadas, segundo a tabela:

Tipo	Declaração	Referência	Bits
inteiro	(unsigned) int	% d	32 bits
inteiro longo	(unsigned) long long	% lld	64 bits
ponto flutuante	(unsigned) float	% f	??
ponto flutuante de dupla precisão	(unsigned) double	% g	??
caractere	char	% c ou % s (strings)	16 bits
ponteiro	<tipo> *	% p	H
nada	void	—	—

- Em C as stringas são criadas como listas de caracteres
- Especiais C++:

Tipo	Declaração	Referência	Bits
string	string	% s	??
booleana	bool	??	1 bit

1.3. Listas

- Seus itens só podem ser do tipo declarado
- São declaradas como:
  - <tipo da lista> lista1[<espaços>];
  - int lista2[] = {1, 3, 5, 7};
- São referenciados como:
  list[0] = 0

1.4. Funções

- Tipos dos argumentos precisam ser fornecidos e o tipo do valor de retorno também (o tipo especificado antes do nome da função), se a função não tiver retorno, colocar void
- São declaradas como:

```
int area(int altura, int largura){
    return (altura * largura);
}
```

1.5. Objetos

- Podem ser class (variáveis automaticamente em privado) ou struct (variáveis automaticamente em público)
- São declarados:

```
class Rectangle {
    int altura;
    int largura;
    public:
        void setDim (int alt, int lar){
            altura = alt;
            largura = lar;}
        int getDim(){
            return {altura, largura};}
}
```

1.6. Expressões

1.6.1. Condicionais

- if
- else if

- else

1.6.2. Loops

- while
- for
- do{(alguma coisa)}while(<condição>);

1.6.3. Outros

- switch

1.7. Comparadores e booleanos

- == e !=
- <= e >=
- < e >
- &&
- ||
- !

1.8. Operadores

- Decimais:
  - + e −
  - * e /
  - % (módulo)
- Binários:
  - & (and)
  - | (or)
  - ∼ (not)
  - ∧ (xor)
  - << (left shift)
  - >> (right shift)

1.9. Bibliotecas

- São importadas do modo:
  - #include <<nome>>
- Mais usadas:
  - <stdio.h> : inputs e outputs para C
  - <iostream> : inputs e outputs para C++
  - <math.h> : matemática
  - <string.h> : strings
  - <vector> : vetor fácil
  - <map> : mapa
  - <queue> : fila fácil
  - <stack> : pilha fácil

1.10. Input e output

- Input:
  - stdio.h: scanf(«referência>", &<variável>); (se a variável for uma string, não colocar o &)
  - iostream: cin >> <variável>;

- Output:
  - stdio.h: printf(«referência> e frase", <variável>);
  - iostream: cout << <variável> << endl;

1.11. Comentários

- Podem ser feitos em uma linha com //, ou em várias com /* */

1.12. Funções e métodos úteis

- pow(base, expoente) (potenciação) (incluída no math.h)
- string.size() (acha o tamanho da string) (incuída no string.h)
- olhar site cplusplus

3. JavaScript

1.1. Indentação

- A indentação é feita por chaves e ponto e vírgula

1.2. Variáveis

- O tipo da variável não precisa ser especificado mas ela precisa ser declarada usando:
  var variável = 42;

1.3. Listas

- Podem mesclar tipos de variáveis
- São declaradas como:
  - lista1 = [] ;
  - lista2 = [0, 0, 0, 0, 0];
  - lista3 = new Array();
- São referenciados como:
  list[0] = 0;

1.4. Funções

- Tipos dos argumentos não precisam ser especificados
- São declaradas como:
  var area = function(altura, largura){
  return (altura * largura);
  }
- Se a função for uma nova classe de objeto, ela deve ser declarada como:
  function Rectangle{
  this.altura = altura; (variável publica)
  this.largura = largura;
  var cor = "branco"; (variável privada)
  }
- Se já existir uma classe e é preciso adicionar algo a mais, usar:
  Rectangle.prototype.area = this.altura * this.largura;
- Se uma classe é uma subclasse de outra, podemos importar os métodos de uma para a outra usando:
  Square.prototype = new Rectangle;

1.5. Objetos

- São declarados:
  var Rectangle = {
  altura: 30,
  largura: 40
  }
- Ou, podem ser declarados:
  var Rectangle = new Object();
  Rectangle.altura = 30;
  Rectangle.largura = 40;

1.6. Expressões

1.6.1. Condicionais

- if
- else if
- else

1.6.2. Loops

- while
- do-while
- for
- for-in

1.6.3. Outros

- switch

1.7. Comparadores e booleanos

- === e !==
- <= e >=
- < e >
- &&
- ||
- !

1.8. Operadores

- + e −
- * e /
- % (módulo)

1.9. Bibliotecas

- Mais usadas:
  - Math : matemática

1.10. Input e output

- Input:
  - prompt(<pergunta>, <resposta padrão>); (retorna a resposta do usuário)
  - confirm(<pergunta>); (retorna true ou false dependendo do botão apertado)
- Output:
  - console.log(<frase>); (console)

- alert(<frase>); (pop-up)

1.11. Comentários

- Podem ser feitos somente em uma linha com //

1.12. Funções e métodos úteis

- Math.random()
- list.length
- string.toUpperCase() e string.toLowerCase()
- object.hasOwnProerty("prop") (verifica se o objeto tem uma propriedade chamada "prop")
- num.toFixed(dig) (limita o número de casas decimais de um "num"float em "dig")
- string.substring(<começo>,<final>); (retorna a substring)

4. HTML e CSS

1.1. Tags

- As tags são a forma de identificação dos comandos e são feitas por <(comando)> (texto) </(comando)>
- As tags tem atributos que são colocados na tag de abertura, como: <p style="font-size: 10px; color: red»(texto vermelho com fonte tamanho 10)

1.2. Layout

- Sempre começar o documento com <!DOCTYPE html> e escrever o código entre <html> e </html>
- Utilizar <body> para o que vai na página e <head> para títulos (body tem atributo style)
- Em <head> colocar <title> para a frase que vai escrita na guia

1.3. Edição de texto

- Título: Pode ser feito de vários tamanhos com <h1> (maior) até <h6> (menor)
- Negrito :
- Itálico:
- Cor da letra : atributo style="color: (cor)"
- Cor de fundo: atrubuto style="background-color: (cor)"
- Local do texto: atributo style="text-align:(right, center ou left)"
- Tamanho da fonte: atributo style="font-size: XXpx"(XX é o tamanho da fonte em pixels)
- Estilo da fonte: atributo style="font-family: (nome)"
- Ver `https://www.w3.org/TR/CSS21/fonts.html#generic-font-families`

1.4. Indentação de frases

- Um novo parágrafo é feito por (frase)
- Uma lista numerada é feita por:

  (primeiro item)
  (segundo item)

(terceiro item)
...

- Uma lista de itens é feita do mesmo jeito da numerada mas substituindo <ol> por
- Todos tem o atributo style

1.5. **Especiais**

- Imagem: (não tem fechamento)
- Hiperlink: (frase)

1.2. **Variáveis**

- O tipo da variável não precisa ser especificado mas ela precisa ser declarada usando:
  var variável = 42;

1.3. **Listas**

- Podem mesclar tipos de variáveis
- São declaradas como:
  - lista1 = [] ;
  - lista2 = [0, 0, 0, 0, 0];
  - lista3 = new Array();
- São referenciados como:
  list[0] = 0;

1.4. **Funções**

- Tipos dos argumentos não precisam ser especificados
- São declaradas como:
  var area = function(altura, largura){
  return (altura * largura);
  }
- Se a função for uma nova classe de objeto, ela deve ser declarada como:
  function Rectangle{
  this.altura = altura; (variável publica)
  this.largura = largura;
  var cor = "branco"; (variável privada)
  }
- Se já existir uma classe e é preciso adicionar algo a mais, usar:
  Rectangle.prototype.area = this.altura * this.largura;
- Se uma classe é uma subclasse de outra, podemos importar os métodos de uma para a outra usando:
  Square.prototype = new Rectangle;

1.5. **Objetos**

- São declarados:
  var Rectangle = {
  altura: 30,
  largura: 40
  }

- Ou, podem ser declarados:
  var Rectangle = new Object();
  Rectangle.altura = 30;
  Rectangle.largura = 40;

1.6. Expressões

1.6.1. Condicionais

- if
- else if
- else

1.6.2. Loops

- while
- do-while
- for
- for-in

1.6.3. Outros

- switch

1.7. Comparadores e booleanos

- === e !==
- <= e >=
- < e >
- &&
- ||
- !

1.8. Operadores

- + e −
- * e /
- % (módulo)

1.9. Bibliotecas

- Mais usadas:
  - Math : matemática

1.10. Input e output

- Input:
  - prompt(<pergunta>, <resposta padrão>); (retorna a resposta do usuário)
  - confirm(<pergunta>); (retorna true ou false dependendo do botão apertado)
- Output:
  - console.log(<frase>); (console)
  - alert(<frase>); (pop-up)

1.11. Comentários

- Podem ser feitos em quantas linhas for preciso com <!– (comentário) –>

1.12. Funções e métodos úteis

- Math.random()
- list.length
- string.toUpperCase() e string.toLowerCase()
- object.hasOwnProerty("prop") (verifica se o objeto tem uma propriedade chamada "prop")
- num.toFixed(dig) (limita o número de casas decimais de um "num"float em "dig")
- string.substring(<começo>,<final>); (retorna a substring)