Capítulo 2 - Tudo se torna programável

**Fluxogramas**

Os fluxogramas são usados em vários setores, incluindo engenharia, ciências físicas e programação de computador, nos quais é necessária uma compreensão completa dos processos ou fluxos de trabalho. Fluxogramas são diagramas que são usados para representar esses processos ou fluxos de trabalho.

Os fluxogramas ilustram a forma como um processo deve funcionar. Os fluxogramas não devem exigir uma terminologia ou símbolos complexos e específicos do setor. Um fluxograma deve ser fácil de entender, sem a necessidade de ser um especialista no campo escolhido.

Os fluxogramas devem mostrar os estados de entrada, as decisões tomadas e os resultados dessas decisões. É importante mostrar os passos que devem ser executados, quando o resultado de uma decisão é sim ou não.

É comum os programadores criarem o primeiro draft de um programa sem uma linguagem de programação específica. Esses programas independentes de linguagem se concentram na lógica, em vez da sintaxe e muitas vezes são chamados de algoritmos. O fluxograma é uma maneira comum de representar um algoritmo. A figura mostra um exemplo de fluxograma.

# **Software do sistema, software de aplicativo e linguagens de computador**

Há dois tipos comuns de software de computador: software do sistema e softwares de aplicativo.

Programas de software de aplicativo são criados para realizar uma determinada tarefa ou conjunto de tarefas. Por exemplo, o Cisco Packet Tracer é um programa de simulação de rede que permite aos usuários modelar redes complexas e perguntar "e se" sobre o comportamento da rede.

O software do sistema funciona entre o hardware do computador e o programa de aplicativo. É o software do sistema que controla o hardware do computador e permite que os programas de aplicativo funcionem. Os exemplos comuns de software do sistema incluem Linux, Apple OSX e Microsoft Windows.

O software do sistema e o software do aplicativo são criados usando uma linguagem de programação. A linguagem de programação é uma linguagem formal desenvolvida para criar programas que comunicam instruções ao hardware do computador. Esses programas implementam algoritmos que são independentes, conjuntos passo a passo de operações que serão realizadas.

Algumas linguagens de computador compilam os programas em um conjunto de instruções de linguagem de máquina. C++ é um exemplo de linguagem de computador compilada. Outras pessoas interpretam essas instruções diretamente, sem primeiro compilá-las na linguagem de máquina. P Python é um exemplo de uma linguagem de programação interpretada. A figura mostra um exemplo de código de Python.

Quando a linguagem de programação é determinada e o processo é inserido em um fluxograma, a criação do programa pode ser iniciada. A maioria das linguagens de computador usam estruturas de programa similares.

# **Variáveis de programação**

As linguagens de programação usam variáveis como blocos dinâmicos para armazenar frases, números ou outras informações importantes que podem ser usadas na codificação. Em vez de repetir valores específicos em vários lugares em todo o código, uma variável pode ser usada. Variáveis podem armazenar o resultado de um cálculo, o resultado de uma consulta de banco de dados ou algum outro valor. Isso significa que o mesmo código funcionará usando diferentes partes de dados, sem precisar ser regravada.

Por exemplo, "x + y = z" é uma expressão de programação. Nesta expressão, x, y e z são as variáveis que podem representar caracteres, strings de caracteres, valores numéricos ou endereços de memória.

Uma variável pode se referir a um valor. Por exemplo, a expressão "a = 10" associa o valor 10 à variável a.

Uma variável também pode representar um local de memória. A expressão "a = 10" representa a que o valor 10 está armazenado em algum local da memória do computador, que é conhecido como 'a'.

As variáveis podem ser classificadas em duas categorias:

* **Variáveis locais** - São as variáveis que estão dentro do escopo de um programa / função / procedimento.
* **Variáveis globais** - São as variáveis que estão no escopo para o tempo de execução do programa. Elas podem ser recuperadas por qualquer parte do programa.

As variáveis permitem que os programadores criem rapidamente uma ampla gama de programas simples ou complexos que informem ao computador para se comportar de forma predefinida.

# 

# **Estruturas de programa básico**

As pessoas transmitem lógica para os computadores por meio de programas. Usando estruturas lógicas específicas, um programador pode preparar um computador para tomar decisões. As estruturas lógicas mais comuns são:

* **IF-THEN** - Essa estrutura lógica permite que o computador tome uma decisão com base no resultado de uma expressão. myVar > 0 é um exemplo de uma expressão. Essa expressão é verdadeira se o valor armazenado na variável myVar for maior que zero. Quando encontrada, a estrutura IF-THEN avalia a expressão fornecida. Se a expressão for falsa, o computador prossegue para a próxima estrutura, ignorando o conteúdo do bloco IF-THEN. Se a expressão for verdadeira, o computador executa a ação associada antes de prosseguir para a próxima instrução do programa. (Figura 1).
* **Loops FOR** – São usados para executar um conjunto específico de instruções, um número específico de vezes, com base em uma expressão. O termo loop vem do fato de que o conjunto de instruções é executado repetidamente. Embora a sintaxe de loops FOR varie de uma linguagem para outra, o conceito permanece o mesmo. Uma variável age como um contador dentro de um intervalo de valores identificado por um mínimo e um máximo. Toda vez que o loop é executado, a variável de contador é incrementada. Quando o contador é igual ao valor máximo definido, o loop é abandonado e a execução passa para a próxima instrução. (Figura 2).
* **Loops WHILE** – São usados para executar um conjunto específico de instruções, enquanto uma expressão for verdadeira. Observe que, muitas vezes, as instruções dentro do loop farão com que expressão seja avaliada como falsa eventualmente. (Figura 3).

**O que é o Blockly?**

O Blockly é uma ferramenta de programação visual criada para ajudar os iniciantes a entender os conceitos de programação. Ao utilizar uma série de tipos de bloqueio, o Blockly permite que um usuário crie um programa sem inserir as linhas de código. Isso é mostrado na figura 1.

O Blockly implementa programação visual, atribuindo diferentes estruturas de programação a blocos coloridos. Os blocos também contêm slots e espaços para permitir que os programadores insiram os valores exigidos pela estrutura. Os programadores podem conectar as estruturas de programação em conjunto, arrastando e anexando os blocos apropriados. Estruturas de programação, como condicionais, loops e variáveis, estão disponíveis para uso.

Criar uma nova variável no Blockly é uma simples questão de arrastar o bloco de variável para o espaço de trabalho e preencher o slot de valor. Também é possível alterar o conteúdo de uma variável à medida que o programa está sendo executado.

A figura 2 mostra uma variável do Blockly.

O Blockly também oferece suporte às funções. Assim como ocorre com as variáveis, o Blockly tem blocos específicos para representar as funções. Assim como ocorre com as variáveis também, basta que os programadores selecionem e arrastem os blocos de função para o espaço de trabalho e preencham os slots necessários.

Observe nas figuras 1 e 2 que o bloco de variável e o bloco de impressão na tela têm uma guia de chanfro na parte inferior e um slot na parte superior. Isso significa que os dois blocos podem ser encaixados para criar uma sequência de programa. O Blockly executará o bloco na parte superior primeiro e, então, prosseguirá para o bloco abaixo dele.

Há outros blocos disponíveis, como um bloco IF-THEN, um bloco WHILE e um bloco FOR. Também existem blocos especificamente para sensores e atuadores.

O Blockly pode ser usado para converter o código de blocos em Python ou JavaScript. Isso é muito útil para os programadores iniciantes.

# **Blockly Games**

O Google fornece uma série de jogos educacionais de código aberto gratuitos, que podem ajudar a aprender a programação. A série é chamada de Blockly Games. Para obter mais informações sobre Blockly Games ou se desejar experimentá-los, clique [aqui](https://blockly-games.appspot.com/). Existem vários níveis a serem concluídos para ajudar a começar. O Blockly pode parecer um brinquedo, mas é uma ótima ferramenta para aperfeiçoar suas habilidades lógicas, que é um dos componentes da linguagem de programação.

# **O que é Python?**

O Python é uma linguagem muito popular que foi projetada para ser fácil de ler e gravar. A comunidade de desenvolvedores do Python agrega valor à linguagem, criando todos os tipos de módulos e disponibilizando esses módulos para outros programadores.

A principal filosofia da linguagem está resumida no documento [The Zen of Python](https://www.python.org/dev/peps/pep-0020/):

* Bonito é melhor que feio
* Explícito é melhor que implícito
* Simples é melhor que complexo
* Complexo é melhor do que complicado
* Contagens de legibilidade

Apesar do fato de que o Python foi criado para ser fácil, ainda existe uma curva de aprendizado. Para facilitar as instruções do Python, o iniciante pode usar o Blockly para melhorar seu entendimento do Python.

Embora linguagens de programação diferentes tenham sintaxe e semântica diferentes, todas elas compartilham da mesma lógica de programação.

Os iniciantes podem usar o Blockly para criar um programa independente de linguagem, exportá-lo como um código do Python e usar esse código recém-criado para saber mais sobre sintaxe, estrutura e semântica do Python.

# **O interpretador do Python**

O Python é uma linguagem interpretada e, portanto, um interpretador é necessário para analisar e executar código Python. O interpretador do Python entende e executa um código Python. Códigos Python podem ser criados em qualquer editor de texto e interpretadores Python estão disponíveis para vários sistemas operacionais. Os desenvolvedores do Python podem criar e implantar os programas do Python em praticamente qualquer sistema operacional. Ferramentas de terceiros, como **Py2exe** e **Pyinstaller**, também podem ser usadas para compactar o código-fonte do Python em um arquivo executável, eliminando a necessidade do interpretador do Python ao executar o código do Python.

Nas máquinas Linux, o interpretador do Python geralmente é instalado em **/usr/bin/python** ou **/usr/bin/python3** (dependendo das versões do Python disponíveis no sistema). Como o novo instalador do Python para Windows, o Python é instalado por padrão no diretório inicial do usuário. Nas máquinas mais antigas do Windows, muitas vezes, o Python é colocado em **C:\PythonXX** (onde XX é a versão do Python). Depois que o interpretador do Python tiver sido instalado, ele vai funcionar mais ou menos como o Linux Shell. Isso significa que quando chamado sem argumentos, ele lê e executa comandos interativamente. Quando chamado com um argumento de nome de arquivo ou com um arquivo como entrada padrão, ele se lê e executa um script desse arquivo.

Para iniciar o interpretador, basta digitar **python** ou **python3** no prompt de shell.

Alguns sistemas antigos ainda estão em execução com uma versão mais antiga do Python 2, mas muitos sistemas novos estão mudando para usar o novo Python versão 3. A versão do Python é impressa na primeira linha, quando o interpretador é iniciado (figura 1). Este curso foi criado com o código do Python 3.

Quando o interpretador do Python é chamado sem argumentos e os comandos são inseridos pelo teclado, o interpretador está no modo interativo. Nesse modo, o interpretador aguarda os comandos. O prompt principal é representado por três sinais de maior (>>>>). Linhas de continuação são representadas por três pontos (...). A continuação é o prompt secundário padrão.

O prompt **>>>** indica que o interpretador está pronto e aguarda os comandos.

As linhas de continuação são necessárias ao inserir o código de várias linhas. A Figura 2 mostra um bloco IF-THEN escrito em Python.

Outra maneira de usar o interpretador é o **python -c command [arg]** ... que executa as instruções no comando. Como, muitas vezes, as instruções do Python contêm espaços ou outros caracteres que são específicos para o shell, recomenda-se inserir o comando inteiro entre aspas simples.

# **Variáveis e instruções básicas no Python**

O interpretador recebe e executa as instruções interativamente.

O interpretador atua como uma calculadora simples. Você pode digitar uma expressão nele e ele irá gravar o valor. A sintaxe da expressão é simples. Os operadores +, -, \* e / funcionam exatamente da mesma maneira que na maioria das outras linguagens (por exemplo, Pascal ou C). Os parênteses (()) podem ser usados para agrupar, como mostrado na figura 1.

O modo interativo do Python implementa a variável especial "\_" para armazenar o resultado da última expressão emitida, como mostrado na figura 2.

Variáveis são rotuladas áreas de memória que são usadas para armazenar dados do programa de tempo de execução. Para atribuir valores às variáveis no Python, use o sinal de igual (=). Os resultados não são exibidos antes do próximo prompt interativo, como mostrado na figura 3.

As tentativas de usar uma variável não definida (nenhum valor atribuído) resultarão em erro, como mostrado na figura 4.

As strings, definidas como uma sequência de caracteres, também podem ser controladas pelo modo interativo. Use o caractere de barra invertida (\) para caracteres de fuga. Por exemplo, uma string usa aspas duplas, mas também precisa usar aspas duplas dentro da string. Se a string for digitada da seguinte forma: **"Eu realmente "preciso" disso"**. O Python ficará confuso e achará que as primeiras aspas duplas dentro da string, na verdade, concluem a string. Se você colocar uma barra invertida (\) antes das aspas duplas dentro da string da seguinte maneira: **"realmente \"preciso\" disso"**, a barra invertida (\) faz com que o Python saia ou ignore o caractere seguinte.

As aspas simples ou aspas duplas podem ser usadas para encapsular as strings, como mostrado na figura 5.

A instrução de impressão imprime o resultado da expressão que recebeu. É diferente de simplesmente gravar a expressão que você deseja gravar (como fizemos anteriormente nos exemplos da calculadora) da forma que controla várias expressões e strings. As strings são impressas sem aspas e um espaço é inserido entre os itens, portanto, você pode formatar os itens corretamente, como mostrado na figura 6.

As funções são uma parte importante de várias linguagens de programação. As funções permitem que um bloco de código receba um nome e seja reutilizado, conforme necessário. A figura 7 define uma função para adicionar dois números e imprimir o resultado.

# **Funções úteis e tipos de dados no Python**

Python é compatível com muitos tipos de dados e de funções úteis. Algumas das funções mais importantes são as seguintes:

**Intervalo()**

A função de intervalo() gera uma lista de números usados geralmente para iterar com loops FOR. A figura 1 mostra exemplos da função de intervalo().

* **intervalo**(**parar**) - Refere-se aos números inteiros a serem gerados, começando com zero.
* **intervalo([iniciar], parar[, passo**] – Este é o número inicial da sequência, o número final na sequência e a diferença entre cada número na sequência.

**Tuplas**

Uma tupla é uma sequência de objetos inalteráveis do Python. As tuplas são sequências separadas por parênteses. A figura 2 mostra exemplos de tuplas.

**Listas**

As listas são uma sequência de objetos alteráveis do Python. As listas podem ser criadas, colocando diferentes valores separados por vírgula entre colchetes. A figura 3 mostra exemplos de listas e como elas podem ser atualizadas.

**Conjuntos**

Os conjuntos são coleções não ordenadas de elementos exclusivos. Os usos comuns incluem testes de associação, remoção de duplicatas de uma sequência e operações matemáticas padrão de computação em conjuntos como intersecção, união, diferença e diferença simétrica. A figura 4 mostra exemplos de conjuntos.

**Dicionários**

Um dicionário é uma lista de elementos separados por vírgulas. Cada elemento é uma combinação de um valor e de uma chave exclusiva. Cada tecla é separada do seu valor por um ponto e vírgula. Todo o dicionário é escrito entre chaves. Elementos do dicionário podem ser acessados, atualizados e excluídos. Também existem várias funções do dicionário incorporadas, como uma função que compara os elementos dentro de dicionários diferentes e outra que fornece uma contagem do número total de elementos dentro de um dicionário. A figura 5 mostra exemplos de dicionários.

# 

# **Estruturas de programação no Python**

Da mesma forma que em outras linguagens, o Python implementa uma estrutura IF-THEN. Os blocos IF-THEN podem ser usados para permitir que o código tome decisões com base no resultado de uma expressão, como mostrado na figura 1.

O código realiza alguns testes e imprime uma mensagem de acordo com os resultados do teste. Observe que o Python também implementa duas subestruturas denominadas ELSE e ELIF. ELSE permite que o programador especifique as instruções para serem executadas se a expressão for falsa. A abreviação de ELSE IF, ELIF, é usada para executar um segundo teste se a primeira expressão for falsa e outro teste for necessário. Pode haver zero ou mais ELIFs e a parte ELSE é opcional.

O loop FOR no Python repete os itens de qualquer sequência (uma lista ou string), na ordem em que aparecem na sequência, como mostrado na figura 2.

O loop WHILE executa um bloco de código se a expressão for verdadeira. O programa mostrado na figura 3 usa um loop WHILE para calcular e imprimir uma subsequência inicial de uma série de Fibonacci, na qual cada número na série é a soma dos dois números anteriores.

A terceira linha contém um operador de atribuição múltipla. As variáveis **a** e **b** recebem os novos valores 0 e 1 em uma única instrução.

O loop WHILE calcula o próximo termo na série Fibonacci, enquanto a condição b < 10 for verdadeira. Como ocorre no C, o Python considera todo número inteiro diferente de zero como verdadeiro e zero como falso. O teste usado na figura é uma comparação simples.

Observe que o corpo do loop está recuado. O recuo é a forma de agrupamento das instruções do Python. No prompt interativo, você deve usar a tecla Tab ou espaço para cada linha recuada. Uma entrada mais complicada do Python deve ser realizada com um editor de texto. Quando for inserida interativamente, uma instrução composta deve ser seguida de uma linha em branco para indicar a conclusão (pois o analisador não pode adivinhar qual será a última linha). Observe que cada linha dentro de um bloco básico deve ser recuada no mesmo valor.

0**Definição da prototipagem**

Prototipagem é o processo de criação de um modelo de trabalho rudimentar de um produto ou o sistema. Para obter a prototipagem na IoT, é útil ter habilidades projetistas, elétricas, físicas/mecânicas (trabalho manual para reunir tudo) e de programação, além de entender como o TCP/IP funciona. Mas você não precisa ser um especialista nessas áreas. De fato, a prototipagem ajuda a obter essas habilidades.

Como a IoT ainda está em desenvolvimento, ainda existem tarefas desconhecidas para descobrir. Essa é uma boa hora para inventar algo que faça parte da IoT. Como a IoT combina pessoas, processos, dados e coisas, não há limites para as invenções que a IoT pode ajudar a criar e incorporar.

Para ver as notícias e as ideias que estão sendo comentadas na IoT, clique [aqui](https://newsroom.cisco.com/internetofthings).

# **Como criar protótipos**

Como você cria protótipos? Existem algumas maneiras de iniciar. Uma equipe do Google usou o "Método rápido de prototipagem" para criar o Google Glass. Clique [aqui](http://ed.ted.com/lessons/rapid-prototyping-google-glass-tom-chi) visualizar um TedTalk sobre este processo.

Obviamente, o Google tem um grande número de recursos para pagar por pessoas e materiais que entram na prototipagem. A maioria de nós precisa de ajuda financeira para colocar as ideias em prática e no protótipo. Para nós, existe o financiamento público. Kickstarter, Indiegogo e Crowdfunder são apenas três dos vários programas de financiamento público online. Clique [aqui](http://vimeo.com/40128933) para assistir ao vídeo Pebble Watch Kickstarter. Esse vídeo online foi usado para gerar doações para ajudar esse grupo de inventores a criar o Pebble Watch.

Que invenção de IoT você vai criar?

**Materiais físicos**

Um bom local para começar é, claro, a Internet. As pessoas vêm trocando ideias por muito tempo, mas a Internet permite trocas de ideias em novo patamar. Pessoas que nunca se encontraram fisicamente podem agora colaborar e trabalhar juntas. Existem vários sites que você pode acessar para se conectar com outros fabricantes.

A Maker Media é uma plataforma global para conectar os inventores uns com os outros para a troca de projetos e ideias. A plataforma também fornece um local onde os fabricantes podem encontrar e comprar produtos para seus projetos. Para obter mais informações, acesse Makezine em [http://makezine.com](http://makezine.com/).

É importante ter habilidades práticas ao trabalhar com determinados materiais; por exemplo, madeira e metal são materiais comuns de prototipagem, mas eles podem ser muito difíceis para um iniciante usar. Você pode se surpreender com o que pode fazer com argila, plástico, papel e fios. Pesquise no Google para obter mais informações ou ideias sobre como trabalhar com os diferentes materiais de prototipagem.

A LEGO Mindstorms tem uma grande comunidade de contribuidores e fãs. Com a LEGO Mindstorms, você pode criar robôs de LEGO e controlá-los usando um aplicativo. Os jogos vêm com tudo o que você precisa para que eles funcionem. Acesse a LEGO Mindstorms em [http://mindstorms.lego.com](http://mindstorms.lego.com/).

Meccano ou Erector Set é um sistema de construção modelo que consiste em tiras de metal, placas, vigas em ângulo, rodas, eixos e engrenagens reutilizáveis com porcas e parafusos para conectar as peças. Ele permite criar protótipos funcionais e dispositivos mecânicos. Acesse o Erector Set em [www.erector.us](http://www.erector.us/).

A impressão em 3D é o processo de fazer um objeto sólido com base em um arquivo do computador de modelo 3D. Uma máquina, chamada de impressora 3D, é conectada ao computador. Várias empresas agora criam e vendem impressoras 3D. Acesse Makerbot em [https://www.makerbot.com](https://www.makerbot.com/).

# **Kits de ferramentas eletrônicas**

Os programas de computador não podem ser executados sem um computador. Embora você possa criar programas para praticamente qualquer computador, algumas plataformas são elaboradas para o iniciante. Abaixo, você encontrará algumas das plataformas mais populares.

Arduino é uma plataforma de computação física de código aberto baseada em uma placa de microcontrolador simples e um ambiente de desenvolvimento para escrever o software para a placa. Você pode desenvolver objetos interativos que precisam de uma variedade de switches ou sensores para controlar luzes, motores e outros objetos físicos. Acesse Arduino em [http://arduino.cc](http://arduino.cc/).

Quando o Arduino não for apropriado para uso como um computador, os baixos requisitos de energia permitirão que ele controle outros dispositivos de forma eficaz.

O Raspberry Pi é um computador do tamanho de um cartão de crédito de baixo custo que pode ser conectado em um monitor de computador ou em uma TV. Você pode operá-lo usando um teclado e um mouse padrão. Ele pode fazer tudo o que um computador faz, desde navegar na Internet e reproduzir vídeos de alta definição a fazer planilhas, processadores de texto e jogar. Acesse Raspberry Pi em [http://www.raspberrypi.org](http://www.raspberrypi.org/).

O Beaglebone é muito semelhante ao Raspberry Pi em tamanho, requisitos de potência e aplicação. O Beaglebone tem mais potência que o Raspberry Pi, por isso, é uma melhor escolha para aplicativos com requisitos de processamento mais altos. Acesse Beaglebone em [http://beagleboard.org](http://beagleboard.org/).

# **Recursos de programação**

A programação é fundamental para a IoT. A criação de um código personalizado é bastante útil para desenvolver uma solução de IoT. Você já aprendeu sobre Blockly e Python. Há muitos outros recursos gratuitos que podem ajudar você a desenvolver suas habilidades de programação.

O OpenCourseWare do MIT (OCW) é uma publicação da Web de quase todo o conteúdo do curso do MIT. Aberto e disponível para o mundo, o OCW é um local excelente para se familiarizar com a programação de computação gratuitamente. Os cursos de programação relacionados ao OCW podem ser encontrados em<http://ocw.mit.edu/courses/intro-programming>.

A Khan Academy é um site educacional sem fins lucrativos criado em 2006 para oferecer uma "educação de alto nível para qualquer pessoa, em qualquer lugar". Palestras relacionadas à programação de computadores podem ser encontradas em<https://www.khanacademy.org/computing/cs>.

A Code Academy é outro recurso excelente. Ela consiste na interatividade para ajudar pessoas a aprenderem como gravar programas de computador. É possível encontrá-la em [http://www.codeacademy.com](http://www.codeacademy.com/).

# **Workshops sobre inventores da comunidade e empreendedorismo**

Talvez você tenha acabado de criar algo muito interessante. E agora? São vários os locais onde você pode obter ajuda para expor sua ideia ou protótipo para outros.

Investigue o que está disponível na comunidade. Verifique com o governo local, escolas e câmaras de comércio para obter informações sobre workshops, aulas e consultoria especializada.

A Internet tem muitos recursos para ajudar na exposição de sua ideia. Um bom exemplo é o Quirky. O Quirky permite que os usuários compartilhem suas ideias. Quando uma ideia é enviada, outros usuários do Quirky podem votar e escolher se desejam ou não apoiar sua ideia. Se uma ideia for boa, ela poderá se tornar um produto real. Você pode saber mais sobre o Quirky em<https://quirky.com/about/>.

# **Resumo**

Este capítulo começou discutindo como aplicar a programação básica para oferecer suporte aos dispositivos de IoT. Fluxogramas são diagramas que são usados para representar processos. Há dois tipos comuns de software de computador: software do sistema e softwares de aplicativo. Programas de software de aplicativo são criados para realizar certas tarefas. O software do sistema funciona entre o hardware do computador e o programa de aplicativo. As variáveis de programação podem ser classificadas em duas categorias:

* **Variáveis locais** - São as variáveis que estão dentro do escopo de um programa / função / procedimento.
* **Variáveis globais** - São as variáveis que estão no escopo para o tempo de execução do programa. Elas podem ser recuperadas por qualquer parte do programa.

As estruturas de lógica mais comuns são IF – THEN, loops FOR e Loops WHILE.

O Blockly é uma ferramenta de programação visual criada para ajudar os iniciantes a entender os conceitos de programação. O Blockly implementa programação visual, atribuindo diferentes estruturas de programação a blocos coloridos.

O Python é uma linguagem muito popular que foi projetada para ser fácil de ler e gravar. O Python é uma linguagem interpretada e, portanto, um interpretador é necessário para analisar e executar código Python. Variáveis são rotuladas áreas de memória que são usadas para armazenar dados do programa de tempo de execução. O Python é compatível com muitas funções úteis e tipos de dados, incluindo intervalo(), tuplas, listas, conjuntos e dicionário. O Python também implementa duas subestruturas denominadas ELSE e ELIF.

Em seguida, o capítulo detalhou a prototipagem. Prototipagem é o processo de criação de um modelo de trabalho rudimentar de um produto ou o sistema. Uma equipe do Google usou o "Método rápido de prototipagem" para criar o Google Glass. A Internet viabiliza as trocas de ideia em um nível totalmente novo. Existem vários sites que você pode acessar para se conectar com outros fabricantes:

[http://makezine.com](http://makezine.com/)

[http://mindstorms.lego.com](http://mindstorms.lego.com/)

[www.erector.us](http://www.erector.us/)

[https://www.makerbot.com](https://www.makerbot.com/)

[http://arduino.cc](http://arduino.cc/)

[http://www.raspberrypi.org](http://www.raspberrypi.org/)

[http://beagleboard.org](http://beagleboard.org/)

<https://blockly-games.appspot.com/>

<http://ocw.mit.edu/courses/intro-programming>

<https://www.khanacademy.org/computing/cs>

[http://www.codeacademy.com](http://www.codeacademy.com/)

<https://quirky.com/about/>