UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA
DIM0124 – PROGRAMAÇÃO CONCORRENTE

Tutorial
Go

Objetivo

O objetivo desta atividade é possibilitar um primeiro contato com a linguagem de programação Go (https://go.dev/).

1 Go: Uma visão geral

Go, também conhecida como *Golang*, é uma linguagem de programação de propósito geral desenvolvida na Google em 2007, tendo sido disponibilizada ao público em código aberto em novembro de 2009. Em dezembro de 2012, Go teve sua primeira versão estável liberada como Go 1, juntamente com sua especificação oficial[1], uma biblioteca padrão e ferramentas para formatação, compilação, verificação e documentação de código fonte. A versão estável corrente de Go é a 1.17.5, liberada em agosto de 2021.

A linguagem de programação Go foi projetada com vistas à construção de sistemas de *software* contemporâneos de larga escala, os quais necessitam ser eficientes, dinâmicos e implantados utilizando arquiteturas de computadores mutliprocessadas e em rede. Nos últimos anos, Go tem sido utilizada tanto pela própria Google quanto por dezenas de organizações do mundo inteiro[2], nos mais diversos domínios de aplicação e complexidades de *software*. Em pouco mais de uma década de existência, Go tornou-se uma linguagem de programação bastante popular principalmente devido a sua simplicidade, desempenho e suporte nativo a programação concorrente e distribuída. Em *rankings* internacionais como o TIOBE Index[3] e o IEEE Spectrum[4] colocam Go dentre as 15 linguagens mais populares entre os desenvolvedores de *software*.

Para alcançar os propósitos originais do projeto da linguagem, Go buscou combinar a leveza, facilidade de uso e expressividade de linguagens interpretadas e dinamicamente tipadas, como JavaScript e Python, com a eficiência e segurança de tradicionais linguagens compiladas e estaticamente tipadas, como C/C++ e Java. A sintaxe de Go assemelha-se bastante à da linguagem de programação C, sendo assim familiar para muitos desenvolvedores, porém mais simplificada e concisa. É importante

[1] *Go Language Specification*: https://go.dev/ref/spec
[2] *Companies currently using Go throughout the world*: https://github.com/golang/go/wiki/GoUsers
[3] *TIOBE Index for January 2022*: https://www.tiobe.com/tiobe-index/
[4] *IEEE Spectrum Top Programming Languages 2021*: https://spectrum.ieee.org/top-programming-languages/

destacar também que, apesar de incorporar alguns conceitos do paradigma de programação orientada a objetos, Go não é considerada uma linguagem orientada a objetos clássica, uma vez que ela não possui as noções de classes e herança, ainda que permita poliformismo.

Uma das principais características de Go é o suporte nativo à concorrência de forma mais facilitada ao programador, por meio de operações de alto nível, contrastando com o esforço significativo requerido para desenvolver, manter e depurar programas concorrentes em linguagens como C++ e Java. Os elementos primordiais para programação concorrente em Go são as chamadas *gorotinas*, fluxos de execução ainda mais leves em termos de uso de memória que as tradicionais *threads*. As gorotinas são criadas e gerenciadas automaticamente pelo ambiente de execução de Go e balanceadas entre os núcleos de processamento disponíveis. Outra característica importante concernente à programação concorrente em Go é a comunicação de dados entre gorotinas utilizando *canais* e não memória compartilhada, minimizando assim problemas como condições de corrida e *deadlocks* que frequentemente ocorrem em outras linguagens de programação com suporte à concorrência.

Um código fonte em Go consiste basicamente de três partes. A primeira delas é a declaração do pacote (*package*) no qual o código fonte em questão está inserido. A filosofia da programação em Go é organizar o código fonte em pacotes, cada um deles agregando em um mesmo diretório um conjunto de arquivos fonte os quais devem ser declarados com o mesmo nome de pacote. Ademais, todo e qualquer programa em Go deve conter um pacote contendo a função principal, que é o ponto de início da execução do programa e a partir do qual o arquivo executável resultante da compilação será gerado. A segunda parte do código diz respeito à importação de pacotes pertencentes à biblioteca padrão da linguagem[5] ou desenvolvidos pelo próprio programador ou por terceiros, os quais contêm a definição de tipos, constantes e funções que podem ser (re)utilizados pelo código fonte em questão. Por fim, a terceira parte refere-se ao restante do código, que pode conter declarações de tipos, variáveis e funções. A título de exemplo, considere o seguinte código fonte em Go, o qual implementa um programa que simplesmente exibe a frase `Olá, mundo!` na saída padrão:

```
1  package main
2
3  import "fmt"
4
5  func main() {
6    fmt.Println("Olá, mundo!")
7  }
```

A declaração de pacote na linha 1 indica que o código fonte em questão contém a função principal (`main`) a ser executada. Dentre os vários pacotes disponíveis como parte da biblioteca padrão de Go, o pacote `fmt`, importado na linha 3, implementa funcionalidades para entrada/saída formatada. Na linha 5, a função `main`, um tipo especial de função que não possui qualquer parâmetro de entrada nem valor de retorno, é declarada como a função principal do programa. Nessa função, a função `Println` do pacote `fmt` é chamada para imprimir a mensagem recebida como argumento na saída padrão (linha 6) e o programa termina com sucesso imediatamente após o término das instruções contidas na função `main`. Note que instruções em Go não são finalizadas com ponto-e-vírgula, como ocorre em várias

[5] *Go Standard Library*: https://pkg.go.dev/std

linguagens de programação.

Para simplesmente executar um código fonte em Go, o usuário deve utilizar, em um terminal, o comando `go run` seguido do nome do arquivo fonte a ser compilado, como neste exemplo:

```
$ go run hello.go
```

No entanto, o comando `go run` realiza a compilação do arquivo fonte sem gerar um arquivo executável visível ao usuário, mas sim um arquivo executável temporário que é destruído imediatamente após o término da execução do programa. Para a obtenção de um arquivo executável propriamente dito a partir da compilação do programa, deve ser utilizado o comando `go build`, seguido do nome do arquivo fonte a ser compilado, como neste exemplo:

```
$ go build hello.go
```

O resultado da execução do comando `go build` é um arquivo executável com o mesmo nome do arquivo fonte compilado (no caso, `hello`) no mesmo diretório. Caso o usuário deseje atribuir outro nome ou localização ao arquivo executável a ser gerado, o usuário deve acrescentar a opção (*flag*) `-o` seguida do nome desejado para o arquivo. Uma vez gerado, o arquivo executável pode ser invocado a partir do terminal como um programa qualquer.

```
$ go build -o helloworld hello.go
```

2 Preparações iniciais

Go disponibiliza distribuições binárias para *download* e instalação nos sistemas operacionais Windows, Linux, FreeBSD e macOS sobre diferentes arquiteturas de processador, bem como possibilita a instalação a partir de seu código fonte. Essas distribuições estão disponíveis no endereço https://go.dev/dl/, juntamente com instruções para instalação.

Após realizar o *download* e instalação da distribuição de acordo com os respectivos sistema operacional e arquitetura de processador de seu computador, você pode opcionalmente realizar a instalação de um ambiente integrado de desenvolvimento (IDE). De acordo com um levantamento realizado com mais de 10 mil desenvolvedores em 2019[6], os ambientes elencados como os mais populares para programação em Go foram Visual Studio Code, GoLand, Vim/Neovim e Emacs, como mostra esta página Web da *Go wiki*: https://github.com/golang/go/wiki/IDEsAndTextEditorPlugins.

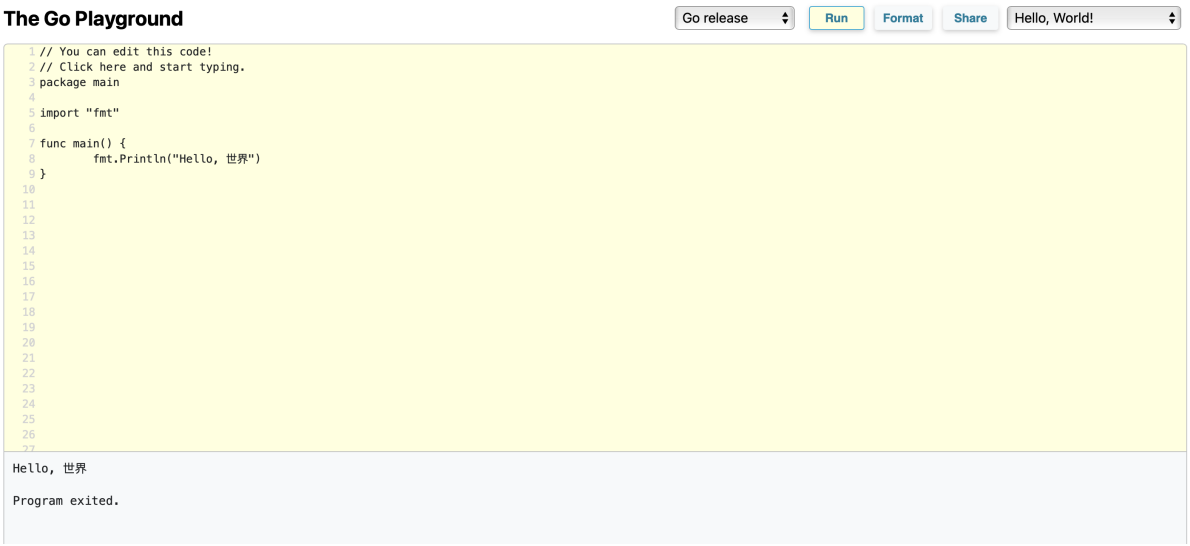
[6] *Go Developer Survey 2019 Results*: https://go.dev/blog/survey2019-results

Stable versions

go1.17.5 ▾

File name	Kind	OS	Arch	Size	SHA256 Checksum
go1.17.5.src.tar.gz	**Source**			**21MB**	3defb9a09bed042403195e872dcbc8c6fae1485963332279668ec52e80a95a2d
go1.17.5.darwin-amd64.tar.gz	Archive	macOS	x86-64	130MB	2db6a5d25815b56072465a2cacc8ed426c18f1d5fc26c1fc8c4f5a7188658264
go1.17.5.darwin-amd64.pkg	**Installer**	**macOS**	**x86-64**	**131MB**	7eb86164c3e6d8bbfba3e4cd30b1f1bd532505594fba2ddf6da6f9838582aab2
go1.17.5.darwin-arm64.tar.gz	Archive	macOS	ARM64	124MB	111f71166de0cb8089bb3e8f9f5b02d76e1bf1309256824d4062a47b0e5f98e0
go1.17.5.darwin-arm64.pkg	**Installer**	**macOS**	**ARM64**	**125MB**	de15daae84a371e3ec45340dbabd657eeca483b35264cd112200ed1026b9e38c
go1.17.5.linux-386.tar.gz	Archive	Linux	x86	101MB	4f4914303bc18f24fd137a97e595735308f5ce81323c7224c12466fd763fc59f
go1.17.5.linux-amd64.tar.gz	**Archive**	**Linux**	**x86-64**	**129MB**	bd78114b0d441b029c8fe0341f4910370925a4d270a6a590668840675b0c653e
go1.17.5.linux-arm64.tar.gz	Archive	Linux	ARM64	98MB	6f95ce3da40d9ce1355e48f31f4eb6508382415ca4d7413b1e7a3314e6430e7e
go1.17.5.linux-armv6l.tar.gz	Archive	Linux	ARMv6	98MB	aa1fb6c53b4fe72f159333362a10aca37ae938bde8adc9c6eaf2a8e87d1e47de
go1.17.5.windows-386.zip	Archive	Windows	x86	115MB	6d7b9948ee14a906b14f5cbebdfab63cd6828b0b618160847ecd3cc3470a26fe
go1.17.5.windows-386.msi	Installer	Windows	x86	101MB	338f42011f44c7e921b4e850a6217aa810526d09dc169bf02530accff47f4e38
go1.17.5.windows-amd64.zip	Archive	Windows	x86-64	143MB	671faf99cd5d81cd7e40936c0a94363c64d654faa0148d2af4bbc262555620b9
go1.17.5.windows-amd64.msi	**Installer**	**Windows**	**x86-64**	**124MB**	93de2b6b56b21940bc061a9c5d68c8b32ca6dff6f947c3d2ac8d3e54728f6d18

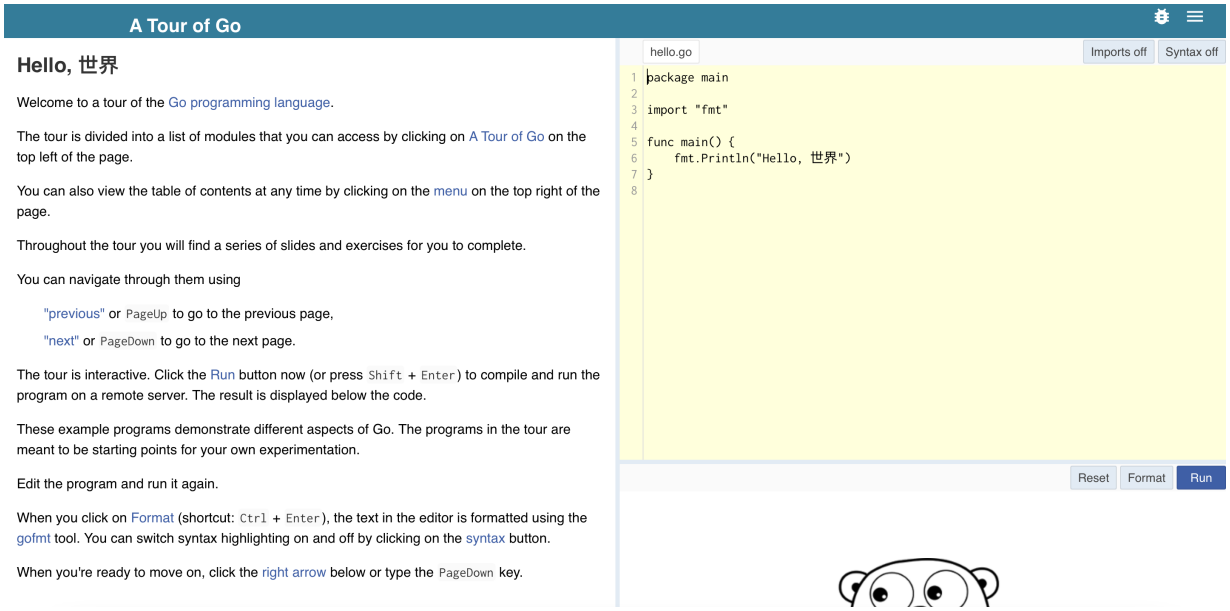
O ecossistema de Go disponibiliza ainda o *Go Playground*, um ambiente que integra editor de texto e compilador para possibilitar ao usuário escrever e executar um código fonte em Go diretamente a partir do seu navegador Web, sem a necessidade de qualquer tipo de instalação ou configuração. O *Go Playground* está disponível para uso a partir do endereço https://go.dev/play/. Após o usuário escrever o seu código fonte no editor de texto, uma vez pressionado o botão *Run*, o ambiente compila o código fonte, executa-o em um servidor Web e exibe o resultado no próprio navegador Web.



Apesar de sua simplicidade e intuitividade, é importante ressalvar que o *Go Playground* possui algumas limitações. As principais delas dizem respeito a dependência de conectividade à Internet, determinismo da execução, impossibilidade de interatividade com o usuário via entrada padrão e limites na utilização de CPU e memória.

3 *A Tour of Go*

A primeira tarefa a ser realizada consiste realizar uma atividade de reconhecimento pelos principais recursos de Go. Esse *tour* é uma introdução interativa à linguagem com diversos exemplos e exercícios simples em três módulos, (i) elementos básicos, (ii) métodos e interfaces e (iii) concorrência. O *tour* pode ser realizado diretamente a partir do navegador Web, a partir do endereço https://go.dev/tour/welcome/1.



Após concluir esse *tour*, espera-se que você esteja apto a dar os primeiros passos no desenvolvimento de seus próprios programas em Go. Para colocar os conhecimentos adquiridos em prática, a próxima tarefa a ser realizada consiste na implementação de um programa para o exercício proposto na Seção 4.

4 Exercício: Uma Calculadora de Geometria Plana e Espacial

A *Geometria* (do grego $\gamma\epsilon\omega\mu\epsilon\tau\rho\iota\alpha$; *geo* = "terra", *metron* = "medida") é um ramo da Matemática que estuda questões relacionadas à forma, tamanho e posição relativa de figuras e a propriedades do espaço. Essa ciência surgiu em diversas culturas da Antiguidade como um conjunto de conhecimentos práticos sobre comprimento, área e volume, partindo das necessidades e observações do ser humano para resolver problemas em agricultura, astronomia, arquitetura e engenharia. Com efeito, conhecimentos em Geometria são aplicados ainda hoje nos mais variados campos do conhecimento humano, tais como Física, Química, Geologia, Astronomia, Engenharia, Biologia, Cartografia e Computação.

Dentre as divisões da Geometria, tem-se a *Geometria Plana* e *Geometria Espacial*. A Geometria Plana refere-se ao estudo das figuras geométricas definidas em um plano de duas dimensões, enquanto que a Geometria Espacial se encarrega do estudo das figuras geométricas (também chamadas de sólidos geométricos) definidas no espaço, ou seja, aquelas que possuem mais de duas dimensões e ocupam um lugar no espaço. As principais figuras geométricas planas são o *triângulo*, o *quadrado*, o *retângulo*

e o *círculo*. Já as principais figuras geométricas espaciais são o *cubo*, a *esfera*, o *cone*, a *pirâmide*, o *paralelepípedo* e o *cilindro*.

Três conceitos são de suma importância para o entendimento das Geometrias Plana e Espacial, a saber, a *área*, o *perímetro* e o *volume*. A área de uma figura geométrica, seja ela plana ou espacial, expressa o tamanho de tal figura sobre uma superfície, de modo que quanto maior a superfície da figura, maior a sua área. O perímetro de uma figura geométrica é definido como a medida do contorno que delimita a figura, sendo resultante da soma das medidas de todos os seus lados. Por fim, o volume corresponde à medida do espaço ocupado por uma figura geométrica. Para encontrar os valores dessas medidas, é importante analisar o tipo da figura (se plana ou espacial) e a forma da figura, isto é, quantos e quais são os lados.

Área, perímetro e volume de figuras geométricas planas e espaciais

A seguir, tem-se a definição das principais figuras geométricas planas e espaciais, bem como as fórmulas utilizadas para calcular as medidas de área, perímetro e volume. É importante notar que, pelo fato de as figuras geométricas planas serem definidas em um plano de duas dimensões, elas não possuem volume.

Área e perímetro das figuras geométricas planas

Figura	**Definição**	**Área**	**Perímetro**
Triângulo*	Figura fechada formada por três lados	$A = \frac{base \times altura}{2}$	$P = lado1 + lado2 + lado3$
Retângulo	Figura fechada formada por quatro lados que formam ângulos retos (90°)	$A = base \times altura$	$P = 2 \times (base + altura)$
Quadrado	Figura fechada formada por quatro lados congruentes (isto é, de medidas iguais) que formam ângulos retos	$A = lado^2$	$P = 4 \times lado$
Círculo	Figura fechada por uma linha curva chamada circunferência	$A = \pi \times r^2$	$P^{**} = 2 \times \pi \times r$

* Para este exercício, você deverá considerar um triângulo equilátero, no qual os três lados são congruentes.

** O perímetro de um círculo é chamado de *comprimento da circunferência*. π é uma constante definida com o valor aproximado de 3,1415 e r é a medida do *raio* do círculo, isto é, a distância entre o centro e a extremidade do círculo.

Área e volume das figuras geométricas espaciais

Figura	**Definição**	**Área**	**Volume**
Pirâmide	Figura composta por uma base poligonal* (triangular, quadrangular, etc.) e um vértice que une as faces laterais da pirâmide	$A = area\_base + area\_lateral^{**}$	$V = \frac{1}{3} \times area\_base \times altura$
Cubo	Figura composta por seis faces quadrangulares	$A = 6 \times aresta^2$	$V = aresta^3$
Paralelepípedo	Figura composta por seis faces, tendo três pares de faces idênticas e paralelas entre si	$A = (2 \times aresta1 \times aresta2) + (2 \times aresta1 \times aresta3) + (2 \times aresta2 \times aresta3)$	$V = aresta1 \times aresta2 \times aresta3$
Esfera	Figura resultante do conjunto de pontos do espaço cuja distância ao centro é igual ou menor que o raio	$A = 4 \times \pi \times r^2$	$V = \frac{4}{3} \times \pi \times r^3$

* Para este exercício, você deverá considerar uma pirâmide com base quadrangular, ou seja, contendo uma base formando um quadrado e quatro faces laterais triangulares.

** A área lateral de uma pirâmide é dada pela soma das áreas de todas as faces laterais triangulares.

Tarefa

Implemente em Go um programa que calcula as medidas de diversas figuras geométricas planas e espaciais. O programa em execução deve apresentar ao usuário uma lista de opções referentes às figuras e, após a escolha de uma dessas opções, deve lhe solicitar que forneça os dados necessários aos cálculos das medidas. Como resultado, o programa deve exibir as respectivas medidas (área, perímetro, volume) da figura escolhida. Note que, para figuras geométricas planas, o programa deve exibir apenas a área e o perímetro; para figuras geométricas espaciais, o programa deve exibir apenas a área e o volume da figura. Para calcular tais medidas, você deve fazer uso das equações matemáticas apresentadas anteriormente, além dos seus próprios conhecimentos matemáticos na área de Geometria.

O programa em execução deve primeiramente apresentar ao usuário uma lista com nove opções, numeradas de 0 a 8, referentes às figuras geométricas planas e espaciais das quais deseja-se obter as medidas. Quando o usuário digitar `0` como opção, o programa deverá ter sua execução finalizada. Caso seja digitada uma opção diferente das disponíveis, o programa deverá emitir uma mensagem informando da entrada inválida e solicitando que o usuário digite uma nova opção.

```
Calculadora de Geometria Plana e Espacial
(1) Triângulo equilátero
(2) Retângulo
(3) Quadrado
(4) Círculo
(5) Pirâmide com base quadrangular
(6) Cubo
```

```
(7) Paralelepîpedo
(8) Esfera
(0) Sair

Digite a sua opção:
```

Após o usuário digitar a opção escolhida, o programa deverá solicitar os dados necessários para o cálculo das medidas de acordo com a figura em questão. Por exemplo, suponha que o usuário tenha escolhido a opção `2`, para cálculo das medidas referentes a um retângulo. Pelo fato de um retângulo ser uma figura geométrica plana, serão calculados apenas a sua área e o seu perímetro, de modo que é necessário informar o tamanho da base e da altura do retângulo.

```
Digite a sua opção: 2
Digite o tamanho da base do retângulo: 5
Digite o tamanho da altura do retângulo: 3
```

Uma vez que o usuário tenha informado os dados necessários aos cálculos, o programa deverá chamar as respectivas funções de cálculos das medidas de acordo com a figura geométrica em questão e exibir os resultados correspondentes. Ainda considerando o exemplo anterior das medidas referentes a um retângulo, o programa exibirá as medidas da área e do perímetro do retângulo.

```
Ârea do retângulo: 15
Perîmetro do retângulo: 16
```

Após a exibição dos resultados das medidas de uma figura geométrica, o programa deverá exibir novamente a lista com as nove opções do menu inicial.

Aplique boas práticas de programação na implementação de seu programa, codificando-o de maneira legível (com indentação de código fonte, nomes consistentes, etc.) e o documentando adequadamente na forma de comentários quando pertinente. Busque ainda desenvolver o seu programa com qualidade, garantindo que ele funcione de forma correta e eficiente. Pense também nas possíveis entradas que poderão ser utilizadas para testar apropriadamente o seu programa. Caso deseje, você poderá ainda modularizar a implementação do seu programa em diferentes pacotes, como preconiza o desenvolvimento de programas em Go.

5 Autoria e política de colaboração

Esta atividade deverá ser feita **individualmente**. O trabalho em cooperação entre estudantes da turma é estimulado, sendo aceitável a discussão de ideias e estratégias. Contudo, tal interação não deve ser entendida como permissão para utilização de (parte de) código fonte de colegas, o que pode caracterizar situação de plágio. Cópia parical ou total de outros colegas ou da Internet serão sumariamente rejeitados e não serão considerados para avaliação.

6 Entrega e avaliação

A realização desta atividade é **opcional**, todavia, caso seja feita com aproveitamento, possibilitará a obtenção de até 2,0 (dois) pontos extras a serem contabilizados para a nota da terceira unidade da disciplina. O(s) código(s) fonte referente(s) ao exercício descrito na Seção 4 deste documento deverá(ão) ser enviado(s) **até as 13h do dia 12 de janeiro de 2022** através da opção *Tarefas* na Turma Virtual do SIGAA. Se for o caso, é possível fornecer, no campo *Comentários* do formulário eletrônico de submissão da tarefa, o endereço de um repositório remoto destinado ao controle de versões ou do compartilhamento no *Go Playground*, porém esta opção não exclui a necessidade de submissão dos arquivos via SIGAA.

7 Alguns livros sobre programação em Go

AIMONETTI, Matt. *Go Bootcamp*: Everything you need to get started with Go. 2016. Disponível em: <http://www.golangbootcamp.com>.

BALBAERT, Ivo. *The way to Go*: A thorough introduction to the Go programming language. USA: iUniverse, 2012.

BATES, Mark; LANOU, Cory; RAYMOND, Tim. *How to code in Go*. 2020. Disponível em: <https://www.digitalocean.com/community/books/how-to-code-in-go-ebook>.

BODNER, Jon. *Learning Go*: An idiomatic approach to real-world. USA: O'Reilly Media, Inc., 2021.

BUTCHER, Matt; FARINA, Matt. *Go in Practice*. USA: Manning Publications Co., 2016.

CHISNALL, David. *The Go programming language*: Phrasebook. USA: Pearson Education, 2012.

D'ANNA, Delio; HAYES, Andrew; HENNESSY, Sam; LEASOR, Jeremy; SOUGRAKPAM, Gobin; SZABÓ, Dániel. *The Go Workshop*: A new, interactive approach to learning Go. United Kingdom: Packt Publishing, 2019.

DONOVAN, Alan A. A.; KERNIGHAN, Brian W. *The Go programming language*. USA: Addison-Wesley, 2015.

DOXSEY, Caleb. *Introducing Go:* Building reliable, scalable programs. USA: O'Reilly Media, Inc., 2016.

FILIPINI, Caio. *Programando em Go*: Crie aplicações com a linguagem do Google. Brasil: Casa do Código, 2014.

GUMUS, Inanc. *Effective Go*: Elegant, efficient, and testable code. USA: Manning Publications Co, 2022.

HARSANYI, Teiva. *100 Go mistakes and how to avoid them*. USA: Manning Publications Co, 2022.

LEE, Wei-Meng. *Go programming language for dummies*. USA: John Wiley & Sons, Inc., 2021.

KENNEDY, William; KETELSEN, Brian; ST. MARTIN, Erik. *Go in Action*. USA: Manning Publications Co., 2016.

RYER, Mat. *Go programming blueprints*. 2. ed. United Kingdom: Packt Publishing, 2016.

SUMMERFIELD, Mark. *Programming in Go*: Creating applications for the 21st Century. USA: Qtrac Ltd./Addison-Wesley, 2012.

VARGHESE, Shiju. *Go recipes*: A problem-solution approach. USA: Apress/Springer Science+Business Media, 2016.

VIVIEN, Vladimir. *Learning Go programming*. United Kingdom: Packt Publishing, 2016.

YOUNGMAN, Nathan; PEPPÉ, Roger. *Get programming with Go*. USA: Manning Publications Co., 2018.