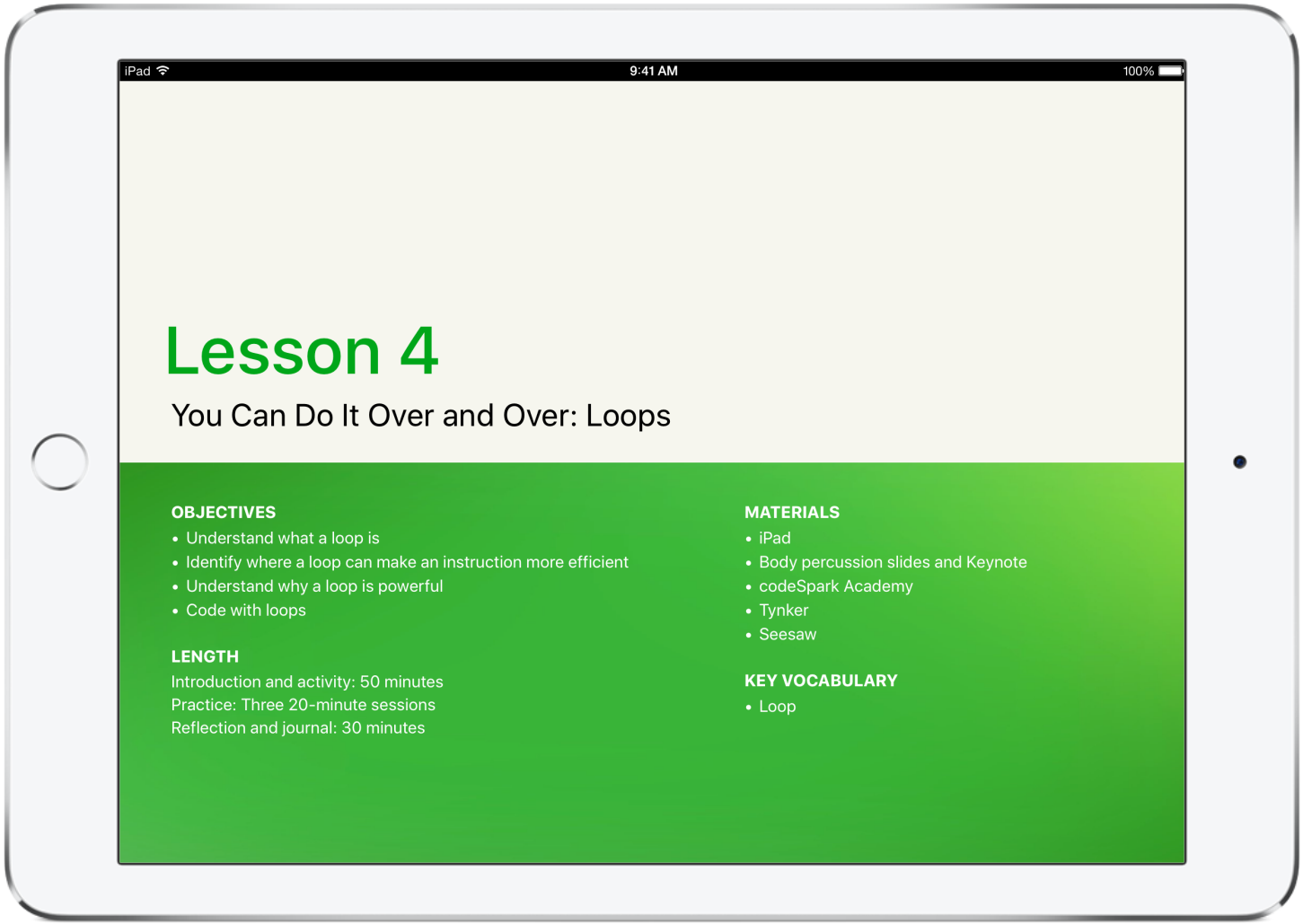
Guia do curso: Comece a Programar

Setembro de 2017

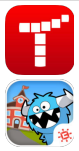
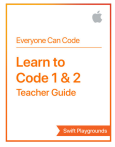
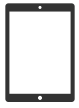
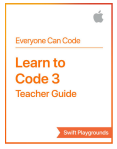
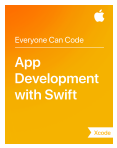


Todo mundo pode programar

A tecnologia tem uma linguagem. Ela se chama código. Acreditamos que dominar essa linguagem é uma habilidade muito importante. A programação ensina a resolver problemas e a trabalhar em equipe de maneiras criativas. Também ajuda a criar apps que colocam suas ideias em prática. Acreditamos que todos deveriam ter a oportunidade de criar algo com potencial de mudar o mundo. Por isso, criamos um novo programa com recursos e ferramentas para que qualquer pessoa possa aprender, formular e ensinar programação.

Currículo do programa Todo mundo pode programar

O programa Todo mundo pode programar conta com diversos que recursos que habilitam os alunos a desenvolver seus primeiros apps, mesmo sem nenhuma experiência em programação. A tabela abaixo apresenta uma visão geral de todos os recursos de ensino e aprendizagem gratuitos disponíveis.

Currículo	Dispositivo	Público	Aplicativo	Pré-requisitos	Visão geral	Materiais de aprendizagem	Recursos de apoio	Horas-aula incluídas
		Do jardim de infância ao 2º ano		Nenhum	Usar apps visuais para explorar conceitos de programação de forma prática e começar a pensar como um programador.	• Aulas do app codeSpark Academy • Curso Tynker Space Cadet	• Comece a Programar 1: Guia do professor	30 horas, incluindo o Guia do professor e as aulas dos apps
		Do 3º ao 5º ano		Nenhum	Explorar os conceitos básicos da programação e usar apps visuais para praticar o pensamento da programação.	• Curso Tynker Dragon Spells	• Comece a Programar 2: Guia do professor	36 horas, incluindo o Guia do professor e as aulas dos apps
		A partir do ensino fundamental II		Nenhum	Aprender os conceitos básicos de programação usando código Swift real.	• App Swift Playgrounds • Aulas dos cursos Aprenda a Programar 1 e 2 • Curso do iTunes U	• Aprenda a Programar 1 e 2: Guia do professor	Até 85 horas, incluindo o Guia do professor e as aulas dos cursos Aprenda a Programar 1 e 2
		A partir do ensino fundamental II		Aprenda a Programar 1 e 2	Expandir as habilidades de programação e começar a pensar como um desenvolvedor de apps.	• App Swift Playgrounds • Aulas do curso Aprenda a Programar 3	• Aprenda a Programar 3: Guia do professor	Até 45 horas, incluindo o Guia do professor e as aulas dos cursos Aprenda a Programar 3
		Ensino médio e superior		Nenhum	Ganhar experiência prática com as ferramentas, técnicas e conceitos necessários para criar um app do iOS do zero.	Livro Introdução ao Desenvolvimento de Apps com Swift e arquivos de projeto	• Introdução ao Desenvolvimento de Apps com Swift: Guia do professor	90 horas
		Ensino médio e superior		Nenhum	Aprender conceitos básicos sobre Swift, UIKit e redes com laboratórios práticos e projetos orientados. Até o final do curso, os alunos serão capazes de criar um app idealizado por eles.	Livro Desenvolvimento de Apps com Swift e arquivos de projeto	• Desenvolvimento de Apps com Swift: Guia do professor	180 horas

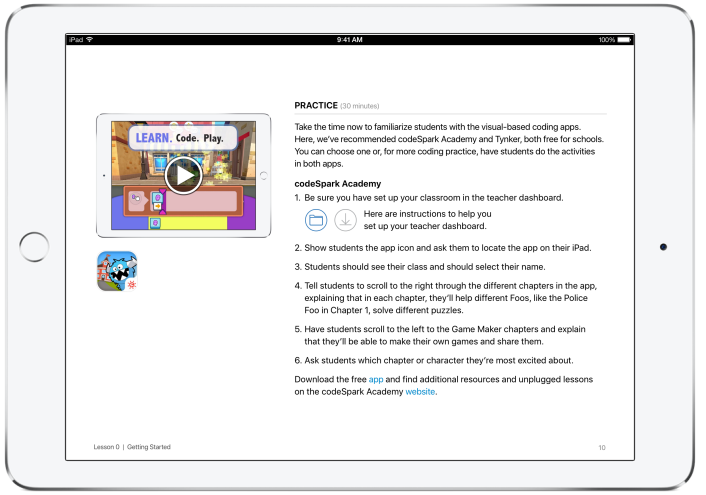
Visão geral

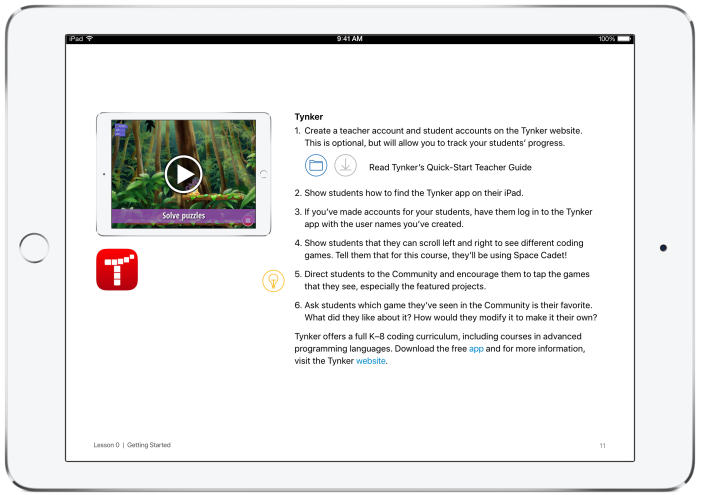
O início da vida estudantil é um ótimo momento para apresentar conceitos de programação como forma de encarar o mundo real e o digital, e também para desenvolver habilidades básicas para o pensamento computacional. Os apps desenvolvidos especificamente para alunos mais novos, como o codeSpark Academy e o Tynker, usam enigmas visuais de programação para desenvolver habilidades de solução de problemas, incentivar a persistência e promover a criatividade. O codeSpark Academy foi criado para alunos de cinco a sete anos. Com uma interface sem textos, o jogo pode ser usado por crianças que ainda não aprenderam a ler, estudantes de inglês e pessoas com dificuldade de leitura. No Tynker, os alunos de cinco a onze anos começam a brincar com blocos visuais e depois avançam para programação baseada em texto, resolvendo enigmas e criando projetos.

Na sala de aula

O Tynker, o codeSpark Academy e as aulas dos Guias do professor do curso Comece a Programar foram desenvolvidos para ajudar você a levar a programação para a sala de aula desde cedo. As aulas destacam os principais conceitos de programação ao demonstrar como a programação é um modo de pensar que pode ser aplicado a outras áreas de aprendizagem e ao dia a dia.

Os Guias do professor oferecem o apoio necessário para ajudar os alunos a resolver os enigmas de programação, independentemente do seu nível de experiência no assunto. Atividades de extensão e de design de apps, perguntas de reflexão, atividades de diário, uma ficha de avaliação e outros materiais para ajudar os alunos a aprofundar a compreensão do conteúdo. As aulas podem ser apresentadas em seções ou de forma corrida. Nos apêndices, há mapas de correlações com um alinhamento preliminar das aulas com o nível 1 dos Padrões de Serviços Provisórios de Ciência da Computação da Associação de Professores de Ciência da Computação (CSTA) K-12.





Principais características

codeSpark Academy

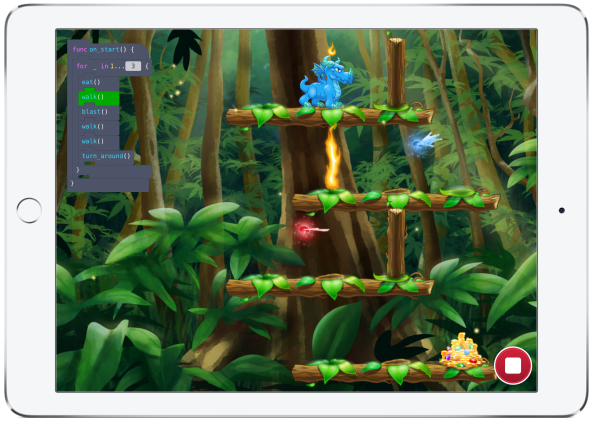
Aprender. Neste jogo sem texto, os alunos resolvem enigmas para aprender conceitos básicos da ciência da computação, como sequenciamento, looping, instruções condicionais e outros.

Criar. Em seguida, os alunos colocam seus conhecimentos em prática programando projetos no Game Maker.

Intuitivo. Não é necessário ter experiência com programação para ensinar, aprender e jogar. O currículo e o painel do professor com relatórios de progresso estão disponíveis de graça em 10 idiomas para os educadores.

Tynker

Ambiente de programação. Os alunos avançam no próprio ritmo por enigmas de programação cada vez mais difíceis para aprender e aplicar conceitos de forma criativa.



Avaliação automática. Um painel do professor avalia o domínio das habilidades pelos alunos, com enigmas, testes e análises de códigos.

Recurso da linguagem Swift. Ao resolverem os enigmas, os alunos podem alternar entre blocos visuais e blocos em Swift. Dessa forma, eles podem conhecer a linguagem Swift e se preparar para programar no futuro.

Comece a Programar: Guias do professor

Arquivos para download. Arquivos de modelo para as atividades dos alunos e apresentações do Keynote para apoiar a aprendizagem em sala de aula.

Gabaritos. Com as soluções dos enigmas do Tynker e do codeSpark Academy, é mais fácil ajudar os alunos que estão com dificuldades para avançar.

Exemplos de trabalhos de alunos. Veja como podem ser as atividades.

Reflexões. Perguntas e sugestões para discussões em aula que ajudam a revisar os conceitos e reforçar a conexão entre o uso dentro e fora do ambiente de programação.

Dicas e exemplos. Ideias para estender ou simplificar as aulas.

Atividades de design de apps. Essas aulas guiam os alunos por um processo de design para criar o conceito e o protótipo da ideia de um app que resolve um problema na sala de aula ou na escola.

Estrutura do curso

Comece a Programar 1

Os alunos começam a pensar como programadores explorando conceitos de programação de um jeito prático e interativo no contexto de situações do dia a dia. Eles aprendem sobre comandos, sequências, loops, eventos e algoritmos. Com trabalhos em grupos, os alunos praticam como prever os resultados da programação e como depurar os códigos deles e dos colegas. Eles também colocam suas habilidades em prática usando apps de programação visual, resolvendo enigmas e desenvolvendo suas próprias criações. As atividades opcionais de design guiam os alunos por um processo de design para criar o conceito e o protótipo da ideia de um app que resolve um problema na sala de aula ou na escola.

Aula 0 — Introdução. Descubra o que os alunos já sabem sobre apps e programação, monte o mural de trabalhos da turma e apresente os principais apps que serão usados nas aulas. Os alunos aprendem as inúmeras funções de uma equipe de design de apps.

Aula 1 — Dá para organizar: introdução ao sequenciamento. Os alunos exploram as sequências do dia a dia, criam uma sequência baseada em uma história conhecida e resolvem enigmas em apps visuais de programação usando sequências simples. Os alunos exploram as diferentes finalidades que os apps cumprem.

Aula 2 — Dá para usar etapas: criação de sequências. Analisando a importância da ordem no sequenciamento de instruções, os alunos aprendem que as mesmas ações podem ser reordenadas para gerar sequências diferentes e criam uma dança louca. Depois, eles descobrem mais comandos e resolvem problemas mais complexos nos apps de programação. Os alunos comparam diferentes apps que ajudam os usuários a aprender novas ideias.

Aula 3 — Dá para escolher: sequenciamento flexível. Os alunos aprendem que a ordem de algumas etapas de uma sequência pode ser flexível. Depois, eles criam as próprias sequências flexíveis, exploram como um enigma pode ser resolvido de várias maneiras e compartilham suas soluções de programação com outros alunos. Os alunos exploram os apps que ajudam os usuários a interagir com outras pessoas na comunidade.

Aula 4 — Dá para repetir: loops. Os alunos identificam loops em contextos do dia a dia e usam a criatividade para criar os próprios loops de percussão corporal. Eles também analisam como os loops são representados na programação e os usam para simplificar os códigos. Os alunos aprendem sobre o design da interface de usuário e como ela pode tornar um app divertido e fácil de usar.

Aula 5 — Dá para consertar: depuração. Os alunos entendem a importância da persistência e da depuração em diversos contextos, analisando e aplicando suas novas habilidades de programação para resolver um desafio e depurar as soluções dos colegas. Eles praticam como prever o resultado dos códigos e identificar bugs quando esses códigos não são executados conforme o planejado. Os alunos fazem uma imersão no design de apps para ajudar a resolver um problema.

Aula 6 — Dá para melhorar: eventos e ações. Os alunos exploram como os eventos podem deixar o app mais interessante e responsivo e aprendem a usá-los para programar. Eles refletem sobre como provocamos eventos no dia a dia e criam um controle remoto robô para praticar a chamada de eventos. Os alunos começam a desenvolver seus próprios apps.

Aula 7 — Dá para fazer tudo se seguir as regras: instruções "if". Os alunos conhecem as instruções condicionais e descobrem como reconhecer as instruções "if" no dia a dia. Eles pensam em instruções "if" que funcionam como as regras de jogos de tabuleiro conhecidos. Depois, eles programam usando instruções "if" para que os códigos respondam melhor às condições do ambiente. Os alunos usam fluxogramas para mostrar como seus apps funcionam.

Aula 8 — Dá para resolver: algoritmos. Combinando tudo o que aprenderam até agora, os alunos criam algoritmos que envolvem uma série de etapas para resolver um problema. Eles começam com receitas simples e avançam até o desenvolvimento e a programação de um jogo de labirinto. Os alunos fazem protótipos de seus apps e concluem um projeto-marco documentando seu processo de design de app.

Estrutura do curso (continuação)

Comece a Programar 2

No curso Comece a Programar 2, os alunos exploram conceitos básicos de programação e começam a pensar como programadores. Além de conhecer algoritmos, funções, loops, instruções condicionais e variáveis, eles aprendem o básico do design de interface de usuário. Os alunos trabalham sozinhos e em grupos para melhorar suas habilidades de programação resolvendo problemas reais, testando os códigos dos colegas e desenvolvendo programas para vários tipos de bots. Eles também praticam essas habilidades no Tynker, resolvendo diferentes problemas e aplicando os conceitos que aprenderam nas atividades em sala de aula. As atividades opcionais de design guiam os alunos por um processo de design para criar o conceito e o protótipo da ideia de um app que resolve um problema na sala de aula ou na escola.

Aula 0 — Introdução. Descubra o que os alunos já sabem sobre apps e programação, mostre um app visual de programação, como o Tynker, e crie os diários digitais dos alunos usando um app como o Seesaw. Os alunos são apresentados a um desafio de design de app.

Aula 1 — Pensar em etapas: solução de problemas com algoritmos. Os alunos descobrem os algoritmos, um conjunto de instruções para resolver um problema ou realizar uma tarefa. No app Tynker, eles desenham um dragão e criam algoritmos para resolver enigmas e aprender habilidades de sequenciamento. Nas atividades em sala de aula, os alunos desenvolvem e testam algoritmos. Os alunos começam a trocar ideias sobre apps que podem ajudar a resolver um problema.

Aula 2 — Pensar em correções: depuração. Os alunos procuram e corrigem erros em seus algoritmos e códigos. No Tynker, eles modificam algoritmos com bugs para criar programações corretas e resolver os enigmas. Os alunos aprendem sobre a função que os teclados desempenham nos apps e como eles podem aplicá-la às ideias de seus próprios apps.

Aula 3 — Pensar em círculos: em busca de loops. Os alunos aprendem que os loops são padrões que se repetem. Depois, eles desenvolvem e testam um algoritmo para criar uma cobra que se movimenta em círculos. No Tynker, eles usam loops para resolver enigmas por meio da identificação de padrões. Os alunos trocam ideias sobre como seu app pode usar o microfone e a câmera integrada.

Aula 4 — Pensar em partes: composição e decomposição. Os alunos precisam inventar um algoritmo para cantar uma "Cup Song". Para isso, eles dividem a coreografia em passos. No Tynker, eles dividem problemas em subproblemas menores para facilitar a resolução. Os alunos pensam em como a tela sensível ao toque pode tornar o app mais interativo.

Aula 5 — Pensar em conjuntos: abstração. Os alunos exploram a similaridade e a generalização categorizando objetos em conjuntos e explicando a lógica usada. No Tynker, eles usam a abstração para identificar semelhanças entre problemas e resolvem enigmas cada vez mais complexos com novas ferramentas de programação. Os alunos pensam em como usar ferramentas como o Bluetooth para se conectar a dispositivos próximos.

Aula 6 — Pensar em padrões: formação de funções. Para criar uma rotina para um bot de comando, os alunos a dividem em funções e trocam algoritmos para testar os resultados previstos e reais. No Tynker, eles usam as funções "name" e "call", e reutilizam conjuntos de instruções para aumentar a eficiência da programação. Os alunos aprendem sobre os tipos de dados aos quais seu app pode se conectar pelo GPS.

Aula 7 — Pensar em detalhes: instruções condicionais. Os alunos embarcam em uma viagem de aventura virtual e usam um conjunto de condições de qualificação com instruções "if" para definir o destino. No Tynker, eles usam instruções "if" para lidar com decisões e alternativas dentro dos enigmas. Os alunos trocam ideias sobre maneiras inovadoras de tornar seus apps únicos.

Estrutura do curso (continuação)

Aula 8 — Pensar em ciclos: loops "while" e loops aninhados.
Nessa aula, os alunos administram uma barraquinha de doces virtual. Para isso, precisam usar loops "while" e loops aninhados e criar algoritmos para que o bot coloque cobertura suficiente nos doces de todos os clientes. No Tynker, os alunos usam loops para reduzir os códigos. Os alunos formam equipes de design de apps e começam a criar um protótipo do próprio app.

Aula 9 — Pensar dentro e fora da caixa: variáveis, entrada e saída.
Usando variáveis, os alunos desenvolvem um algoritmo para um campeonato de poesia falada e criam uma música ou um rap que vai mudando conforme a reação do público. No Tynker, eles usam variáveis para resolver enigmas mais complexos, aplicando todas as habilidades de programação que aprenderam até agora. Os alunos conduzem entrevistas com usuários para ajudá-los a definir um público para seu app.

Aula 10 — Pensar na prática: design de interface. Os alunos analisam as características de um bom design e criam uma placa para a escola. Na atividade do Tynker, eles usam todas as habilidades que aprenderam e concluem o curso Comece a Programar 2. Os alunos aprendem sobre a interface e experiência do usuário e criam um quadro de humor para o design de seu app. Eles criam uma apresentação do app no projeto-marco.

Outras informações

Baixe os recursos dos cursos Comece a Programar

- Tynker
- codeSpark Academy
- Comece a Programar 1
- Comece a Programar 2

Baixe os recursos do Swift Playgrounds

- Aprenda a Programar 1 e 2: curso do iTunes U
- Aprenda a Programar 1 e 2: Guia do professor
- Aprenda a Programar 3: Guia do professor
- App Swift Playgrounds

Baixe os guias do curso Desenvolvimento de Apps com Swift

- Introdução ao Desenvolvimento de Apps com Swift
- Introdução ao Desenvolvimento de Apps com Swift: Guia do professor
- Desenvolvimento de Apps com Swift
- Desenvolvimento de Apps com Swift: Guia do professor

Outros recursos

- Saiba mais sobre o programa Todo mundo pode programar.
- Saiba mais sobre a linguagem Swift.
- Saiba mais sobre o Xcode.
- Converse com outros educadores nos Fóruns de desenvolvedores da Apple.
- Saiba mais sobre o codeSpark Academy.
- Saiba mais sobre o Tynker.

Alinhamento do currículo: Comece a Programar 1

As aulas do curso Comece a Programar 1 estão alinhadas aos conceitos de algoritmos e programas do nível 1 dos Padrões de Serviços Provisórios de Ciência da Computação da Associação de Professores de Ciência da Computação (CSTA) K-12 de 2016 para o jardim de infância ao 2º ano.

Alinhamento do Comece a Programar 1 com o nível 1 dos Padrões de Ciência da Computação da CSTA K–12 para o jardim de infância ao 2º ano

Comece a Programar 1 – Capítulos	Padrão da CSTA	1A-A-7-1 Conteúdo dos créditos	1A-A-5-2 Construir programas	1A-A-5-3 Desenvolver documentos	1A-A-4-4 Representar dados	1A-A-3-5 Decompor	1A-A-3-6 Categorizar itens	1A-A-3-7 Algoritmos	1A-A-6-8 Analisar e depurar
	Correlação geral		●	●	●	●	●	●	●
	Dá para organizar: introdução ao sequenciamento		●	●	●	●	●	●	●
	Dá para usar etapas: criação de sequências		●	●	●	●	●	●	●
	Dá para escolher: sequenciamento flexível		●	●	●	●	●	●	●
	Dá para repetir: loops		●	●	●	●	●	●	●
	Dá para consertar: depuração		●	●	●	●		●	●
	Dá para melhorar: eventos e ações		●	●	●	●		●	●
	Dá para fazer tudo se seguir as regras: instruções "if"		●	●	●	●		●	●
	Dá para resolver: algoritmos		●	●	●	●		●	●

Legenda: ● Correlação geral ● Alinha-se ao padrão

Alinhamento do currículo: Comece a Programar 2

As aulas do curso Comece a Programar 2 estão alinhadas aos conceitos de algoritmos e programas com o nível 1 dos Padrões de Serviços Provisórios de Ciência da Computação da Associação de Professores de Ciência da Computação (CSTA) K-12 de 2016 para o 3º ao 5º ano.

Alinhamento do Comece a Programar 2 com o nível 1 dos Padrões de Ciência da Computação da CSTA K–12 para o 3º ao 5º ano

Comece a Programar 2 – Capítulos	Padrão da CSTA	**1B-A-2-1** Estratégias de colaboração	**1B-A-7-2** Citação e documentação	**1B-A-5-3** Planejamento	**1B-A-5-4** Construir programas	**1B-A-5-5** Operações matemáticas	**1B-A-3-6** Decompor	**1B-A-3-7** Algoritmos	**1B-A-6-8** Analisar e depurar
	Correlação geral	●	●	●	●	●	●	●	●
	Pensar em etapas: solução de problemas com algoritmos	●		●	●		●	●	●
	Pensar em correções: depuração	●	●	●	●		●	●	●
	Pensar em círculos: em busca de loops	●		●	●		●	●	●
	Pensar em partes: composição e decomposição	●		●	●		●	●	●
	Pensar em conjuntos: abstração	●		●	●			●	●
	Pensar em padrões: formação de funções	●	●	●	●		●	●	●
	Pensar em detalhes: instruções condicionais	●	●	●	●		●	●	●
	Pensar em ciclos: loops "while" e loops aninhados	●		●	●	●	●	●	●
	Pensar dentro e fora da caixa: variáveis, entrada e saída	●		●	●		●	●	●
	Pensar na prática: design de interface	●	●	●	●		●		●

Legenda: ● Correlação geral ● Alinha-se ao padrão

Os recursos estão sujeitos a mudanças. Alguns recursos podem não estar disponíveis em todas as regiões ou idiomas.