

Elaboração de Planos de Ensino-Aprendizagem

Ajuda	Encaminha Plano Para Aprovação	Copia Conteúdo de Outro Plano (2021/2)	Copia Conteúdo de Outro Plano (ano/semestres anteriores)
-----------------------	--	--	--

Fluxo

Situação	Data	Executor	Descrição
Disponível para elaboração	09-07-2021 09:11:18	Wilson Guilherme Lobe Junior	
Em elaboração	13-07-2021 11:21:01	Dalton Solano dos Reis	

Informações FURB

<p>Plano de Desenvolvimento Institucional - PDI</p> <p>Missão: promover o ensino, a pesquisa e a extensão, fomentando o desenvolvimento socioeconômico sustentável e o bem-estar social.</p> <p>Visão: ser uma Universidade pública, reconhecida pela qualidade da sua contribuição na vida regional, nacional e global.</p> <p>Valores: transparência; participação; valorização dos discentes e dos servidores; formação integral do ser humano; democracia; ética; pluralidade; desenvolvimento social e sustentável; manutenção da sua identidade e tradição; respeito à natureza e a todas as formas de vida.</p>
<p>Projeto Pedagógico Institucional - PPI</p> <p>Princípios do Ensino: Democracia e Direitos Humanos; ética e Cidadania ambiental; relações étnico-sociais; formação Crítica.</p> <p>Diretrizes para o Ensino: aprendizagem como foco do processo; educação geral; flexibilização; tecnologias digitais, internacionalização.</p>

Identificação

Ano/Semestre:	2021/2	Turma:	CMP.0166.00.002
Nome da Disciplina:	Introdução à Programação		
Centro:	Centro de Ciências Exatas e Naturais		
Departamento:	Departamento de Sistemas e Computação		

Carga Horária

Créditos			Carga Horária semestral		
Teóricos:	6	Práticos:	1	Total:	7
				Teórica:	108
				Prática:	18
				Total:	126

Cursos

20 - Ciência da Computação (Noturno)		Currículo: 2019/1 Fase(s): 1/A	
Objetivo do curso O curso de Ciência da Computação da Universidade Regional de Blumenau tem como objetivo formar um profissional com conhecimento científico e base sólida em computação, atendendo de forma proativa e ética às demandas da comunidade regional.			
Objetivo geral da disciplina Identificar problemas que tenham solução algorítmica, utilizando um método para resolução dos problemas computacionais e implementando as soluções por meio de programas escritos em uma linguagem de programação.			
Ementa Fundamentos da programação de computadores. Construção de algoritmos. Introdução a linguagem de programação. Comandos de controle de fluxo: seleção, repetição e sub-rotinas. Tipos estruturados: vetores. Introdução a OO: classes e objetos, atributos e métodos.			
Pré-Requisitos			
Nome da Disciplina		Código da disciplina	Tipo

126 - Sistemas de Informação (Noturno)		Currículo: 2020/1 Fase(s): 1/A	
Objetivo do curso O objetivo do curso de Sistemas de Informação da Universidade Regional de Blumenau é formar profissionais capazes de desenvolver e aplicar as tecnologias da informação na solução de problemas das organizações, atendendo de forma proativa e ética às demandas da comunidade regional.			
Objetivo geral da disciplina			
Ementa Fundamentos da programação de computadores. Construção de algoritmos. Introdução a linguagem de programação. Comandos de controle de fluxo: seleção, repetição e sub-rotinas. Tipos estruturados: vetores. Introdução a OO: classes e objetos, atributos e métodos.			
Pré-Requisitos			
Nome da Disciplina		Código da disciplina	Tipo

Unidades

	Unidades e Subunidades	Objetivos Específicos	Procedimentos Metodológicos	Instrumentos e Critérios de Avaliação
editar excluir	1. FUNDAMENTOS DA PROGRAMAÇÃO DE COMPUTADORES 1.1. Solução de Problemas 1.2. Técnicas para representação da solução	Compreender o que é a solução de problemas computacionais. Identificar os elementos básicos para a solução de problemas computacionais. Conhecer técnicas de representação para a solução de problemas computacionais.	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, discussões em grupo em sala de aula e autoestudo.	Instrumento: Exercícios individuais ou em grupo. Prova Individual (Prova 1). Critério: Compreensão sobre o que é a programação de computadores, e avaliação do autoestudo.
	2. CONSTRUÇÃO DE ALGORITMOS 2.1. Dados e Tipos 2.2. Comandos e Instruções	Compreender os principais elementos de um algoritmo. Identificar dados e definir seus tipos. Conhecer as instruções básicas de algoritmos. Representar a solução de algoritmos por meio de	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, rotação por estações, discussões em grupo em sala de aula e autoestudo.	Instrumento: Prova Individual (Prova 1) Exercícios parciais individuais ou em grupo.

editar excluir		português estruturado.		Critérios: Capacidade de interpretação e resolução de problemas, e avaliação do autoestudo. Coerência e lógica das especificações realizadas. Colaboração no trabalho em equipe.
editar excluir	3. INTRODUÇÃO A LINGUAGEM DE PROGRAMAÇÃO 3.1. Introdução a uma IDE 3.2. Características da linguagem de programação 3.3. Tipos de dados 3.4. Palavras reservadas 3.5. Operadores 3.6. Comandos de entrada e saída 3.7. Método main e conceitos de subprograma (sub-rotinas e funções) 3.8. Passagem de parâmetros por valor e referência 3.9. Retorno da função	Utilizar uma IDE. Conhecer as características e recursos básicos de uma linguagem de programação. Conhecer a sintaxe e semântica básica da linguagem. Conhecer as principais estruturas da linguagem. Implementar soluções de problemas simples em uma linguagem de programação. Conhecer os conceitos de definição de subprogramas. Entender a diferença entre passagem de parâmetro por valor e referência. Entender o uso do retorno das funções.	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, rotação por estações, plataforma de exercícios personalizada, discussões em grupo em sala de aula e autoestudo.	Instrumentos: Prova Individual (Prova 1) Exercícios de Implementação de Programas Critérios: Capacidade de interpretação e resolução de problemas, e avaliação do autoestudo. Coerência e lógica das especificações realizadas. Implementação correta dos problemas propostos. Colaboração no trabalho em equipe.
editar excluir	4. COMANDOS DE CONTROLE DE FLUXO: SELEÇÃO 4.1 Simples: se (if) 4.2 Encadeada: se-senão (if - else) 4.3 Múltipla: escolha (switch - case)	Compreender as características e aplicações adequadas para o uso dos comandos de controle de fluxo. Implementar soluções de problemas utilizando os comandos de seleção.	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, rotação por estações, plataforma de exercícios personalizada, discussões em grupo em sala de aula e autoestudo.	Instrumentos: Prova Individual (Prova 1) Exercícios de Implementação de Programas Critérios: Capacidade de interpretação e resolução de problemas Coerência e lógica das especificações realizadas. Implementação correta dos problemas propostos. Colaboração no trabalho em equipe, e avaliação do autoestudo.
editar excluir	5. COMANDOS DE CONTROLE DE FLUXO: REPETIÇÃO 5.1. Enquanto (while) 5.2. Para (for) 5.3. Faça Enquanto (do - while)	Compreender as características e aplicações adequadas para o uso dos comandos de controle de fluxo. Implementar soluções de problemas utilizando os comandos de repetição.	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, rotação por estações, plataforma de exercícios personalizada, discussões em grupo em sala de aula e autoestudo.	Instrumentos: Prova Individual (Prova 2) Exercícios de Implementação de Programas Critérios: Capacidade de interpretação e resolução de problemas. Coerência e lógica das especificações realizadas. Implementação correta dos problemas propostos. Colaboração no trabalho em equipe, e avaliação do autoestudo.
editar excluir	6. TIPOS ESTRUTURADOS 6.1. Características dos tipos estruturados 6.2. Vetores	Entender o conceito de tipos estruturados. Identificar o uso adequado dos tipos estruturados em diferentes situações. Compreender a aplicação de vetores. Implementar programas utilizando vetores.	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, aprendizagem baseada em projetos, discussões em grupo em sala de aula e autoestudo.	Instrumentos: Prova Individual (Prova 2) Exercícios de Implementação de Programas Critérios: Capacidade de interpretação e resolução de problemas. Coerência e lógica das especificações realizadas. Implementação correta dos problemas propostos. Colaboração no trabalho em equipe, e avaliação do autoestudo.
editar excluir	7. INTRODUÇÃO A ORIENTAÇÃO A OBJETOS 7.1. Classes e objetos 7.2. Atributos 7.3. Mensagens e métodos 7.4. Encapsulamento 7.5. Construtores	Conhecer os conceitos fundamentais da orientação a objetos. Entender o conceito de classes e objetos. Identificar atributos e métodos em uma classe, definido seus tipos. Construir assinaturas de métodos. Descrever o algoritmo de um método em passos gerais. Compreender o encapsulamento e os qualificadores de acesso de atributos e métodos. Entender e definir métodos construtores. Compreender e resolver problemas básicos usando orientação a objetos.	Aulas expositivas dialogadas, sala de aula invertida com videoaulas, exercícios com gamificação, rotação por estações, aprendizagem baseada em projetos, discussões em grupo em sala de aula e autoestudo.	Instrumentos: Exercícios individuais ou em grupo. Projeto em grupo. Critérios: Capacidade de interpretação e resolução de problemas. Coerência e lógica das especificações realizadas. Criatividade na elaboração do projeto. Colaboração no trabalho em equipe, e avaliação do autoestudo.

[Acrescentar nova unidade](#)

Documentos Recomendados

Básico	
editar excluir	DEITEL, Paul J; DEITEL, Harvey M. Java: como programar .8. ed. São Paulo: Pearson, 2010. xxix, 1144 p, il.
editar excluir	FURGERI, Sérgio. Java 7: ensino didático .2. ed. rev. e atual. São Paulo : Érica, 2012. 320 p, il.
editar excluir	JANDL JÚNIOR, Peter. Java: guia do programador : atualizado para Java 7. 2. ed. São Paulo : Novatec, 2013. 640 f, il.
editar excluir	SANTOS, Rafael. Introdução à programação orientada a objetos usando JAVA .2. ed. Rio de Janeiro : Elsevier, 2013. 313 p, il.
editar excluir	SOUZA, Marco Antonio Furlan de. Algoritmos e lógica de programação . São Paulo : Pioneira Thomson, 2005. xxiii, 214 p, il.
Complementar	
editar excluir	ANSELMO, Fernando. Aplicando lógica orientada a objetos em Java .2. ed. atual. e ampl. Florianópolis : Visual Books, 2005. 178 p, il.
editar excluir	BORATTI, Isaías Camilo. Programação orientada a objetos em JAVA . Florianópolis : Visual Books, 2007. 310 p, il.
editar excluir	CARBONI, Irenice de Fátima. Lógica de programação . São Paulo : Pioneira Thomson Learning, 2003. 240 p, il.
editar excluir	HORSTMANN, Cay S. Big Java . Porto Alegre : Bookman, 2004. xi, 1125 p, il. , 1 CD-ROM.
editar excluir	LOPES, Anita; GARCIA, Guto. Introdução à programação : 500 algoritmos resolvidos. Rio de Janeiro : Elsevier : Campus, 2002. 469 p, il. , 1 CD-ROM.
editar excluir	MANZANO, José Augusto N. G; OLIVEIRA, Jayr Figueiredo de. Algoritmos: lógica para desenvolvimento de programação . Sao Paulo : Erica, 1996. 265p, il.
editar excluir	PUGA, Sandra. Lógica de programação e estruturas de dados com aplicações em Java . São Paulo : Pearson Education : Prentice Hall, 2003. xv, 254 p, il.
editar excluir	SCHILD, Herbert; HOLMES, James. A arte do Java . Rio de Janeiro : Elsevier : Campus, c2003. xvi, 382 p, il.
editar excluir	VILARIM, Gilvan de Oliveira. Algoritmos: programação para iniciantes .2. ed. Rio de Janeiro : Ciência Moderna, 2004. xiv, 270 p, il.
editar excluir	XAVIER, Gley Fabiano Cardoso. Lógica de programação .7. ed. São Paulo : SENAC, 2004. xxv, 378 p, il. 1 CD-ROM. (Nova série informática).
Eletrônico	
editar excluir	Java com VSCode Java in Visual Studio Code
editar excluir	Java no VSCode Getting Started with Java in VS Code
editar excluir	OpenJDK Documentation Documentação do OpenJDK (Java).
editar excluir	VSCode IDE Visual Studio Code da Microsoft.
editar excluir	3 - URI Online Judge Problems & Contests (www.urionlinejudge.com.br/judge/pt/login) URI - Plataforma para resolução de problemas
editar excluir	4 - GUI (www.guj.com.br/) Comunidade de desenvolvedores Java
editar excluir	5 - Canal Java - DevMedia (www.devmedia.com.br/java/) Artigos e dicas da linguagem Java
editar excluir	6 - Stack Overflow (pt.stackoverflow.com/) Plataforma de perguntas e respostas para programadores.
editar excluir	github.com/dalton-reis/disciplinaIpNot REIS, Dalton S. dos. Introdução à Programação: notas de aula (GitHub). Blumenau, 2019. Disponível em: . Acesso em: 23 Jul. 2021.

Incluir novo(s) documento(s) ao Plano:

[tipo "básico" ou "complementar" a partir do acervo da Biblioteca Central](#)
[sugerir aquisições](#)
[tipo "eletrônico"](#)

Dados Complementares do Professor (utilize o botão "Salvar" no final da página após preencher este campo)

E-mail/MS-Teams: dalton@furb.br
 Material disciplina: AVA3
 Repositório GIT: <https://github.com/dalton-reis/disciplinalpNot>
 Grupo pesquisa: <http://gcg.inf.furb.br> e <http://tecedu.inf.furb.br> / <https://www.zotero.org/daltonreis>

Procedimentos de Avaliação (utilize o botão "Salvar" no final da página após preencher este campo)

- A média final será calculada pela seguinte fórmula:

$$\text{Média Final} = \text{Prova 1} * 0.2 + \text{Prova 2} * 0.2 + \text{Projeto FINAL} * 0.4 + \text{Média Aritmética dos Demais Trabalhos Parciais} * 0.2$$

- As provas serão individuais realizadas durante as aulas.

- O projeto FINAL será desenvolvido em equipe mas avaliado individualmente (os alunos da mesma equipe podem ter notas diferentes dependendo do resultado da avaliação individual). O cenário a ser desenvolvido FINAL será definido pelo professor e pode envolver todo o conteúdo da disciplina.

- Nos trabalhos parciais serão propostos quiz para revisão de conceitos e exercícios de programação usando sobretudo o site URI. Também serão realizados exercícios em outras plataformas solicitadas pelo professor.

- Em caso de verificação de cópia de provas, projeto ou trabalhos, a nota questão será ZERADA, tanto para o aluno que copiou, quanto para o que deixou copiar. Todos os trabalhos desenvolvidos na disciplina para serem apresentados ao professor para arguição, e constatado que o aluno não desenvolveu o referido trabalho, a nota questão será ZERADA.

- Os trabalhos deverão ser entregues até a data estipulada pelo professor em aula, podendo haver trabalhos sem data marcada previamente, definidos durante as aulas.

- O cronograma detalhado da disciplina encontra-se no AVA3.

De acordo com o regimento geral da FURB, artigo 66, o aluno que faltar a alguma atividade de avaliação poderá requerer ao professor nova oportunidade em até 5 (cinco) dias úteis, mediante expressa justificativa do professor decidirá a forma de recuperação da nota.

Observações (utilize o botão "Salvar" no final da página após preencher este campo)

Ferramentas básicas para a utilização da disciplina:

- Visual Studio Code com as extensões Live Share e Java;
- Java OpenJDK 11 ou superior;
- Ferramenta de versionamento de código GIT para uso no GitHub e VSCode.

Não é admitida, sob hipótese alguma, cópia de trabalhos ou "compartilhamento de código" com colegas. Todos os trabalhos nos quais o professor concluir que houve cópia (mesmo que parcial) receberão nota zero e possibilidade de reavaliação dos trabalhos. Os alunos devem tomar os devidos cuidados para proteger seu código contra cópias para reuso em outros trabalhos. Caso venha a usar um repositório, use ele em modo privado. Na modalidade de aulas presenciais, o professor pode pedir para desligar o computador (pessoal e/ou do laboratório) durante as aulas (teóricas e/ou práticas), caso julgue necessário. Na modalidade de aulas remotas, o professor pode pedir para o aluno manter a câmera do MS-Teams aberta para validar sua presença na aula. Bem como, pode pedir para ligar/desligar o microfone para diminuir o ruído ambiente.

Mais referências bibliográficas serão disponibilizadas pelo professor durante o desenvolvimento da disciplina.

Toda comunicação digital será feita por chat no MS-Teams ou e-mail, usando o e-mail institucional da Furb do aluno (nickname_do_aluno@furb.br).

Plano de ensino adaptado, em caráter excepcional e transitório, para substituição do ensino presencial pelo ensino não presencial, enquanto durar a pandemia do novo coronavírus (COVID-19) em atenção à:

- *Lei Ordinária Estadual de SC nº 18.032, de 08 de dezembro de 2020, que considera como essenciais durante a pandemia de COVID-19 as atividades educacionais presenciais nas unidades das redes pública e municipal, estadual e federal, relacionadas à educação infantil, ensino fundamental, nível médio, Educação de Jovens e Adultos (EJA), ensino técnico, ensino superior e afins;
- *Portaria nº 476/2021 de 06 de maio de 2021, que autoriza as atividades educacionais presenciais e determina rotinas e normas de biossegurança para os estabelecimentos de ensino;
- *Decreto Estadual nº 1.153, de 15 de fevereiro de 2021, que alterou o Decreto Estadual nº 1.003/2020, que autoriza as atividades educacionais presenciais, em qualquer nível de ensino; - os Decretos Municipais nº 1.003/2020 e nº 1.004/2020;
- *Parecer CEE/SC nº 400, de 08/12/2020 que recepciona o Parecer CNE/CP nº 15/2020, combinado com a nova redação dada pelo Parecer CNE/CP nº 19/2020, resultando, assim, na Resolução CNE/CP nº 2/2021 aplicabilidade ao Sistema de Educação de Santa Catarina;
- *Instrução Normativa PROEN nº 03/2020, de 06/06/2020;
- *Nota Oficial FURB 24/2021.

As atividades desta disciplina serão desenvolvidas no modelo Onlife, transmitidas ao vivo, permitindo a interação com os alunos que estiverem acompanhando presencialmente ou de forma remota, além de serem disponibilizadas ao estudante assistir aos conteúdos posteriormente.

Salvar



DTI - Seção de Desenvolvimento de Sistemas [27-Jul-2021 18:31:29]
[Início](#) [Meus Planos de Ensino na Graduação](#) [Sair](#)