

Instruções para Entrega da Unidade 1

Introdução a Técnicas de Programação (2025.2)



Resumo da Entrega

Data limite: 30/09/2025 às 23:59

Plataforma: SIGAA - Tarefa "Entrega U1"

Formato: **UM ÚNICO ARQUIVO .zip** contendo todos os itens



O que deve ser entregue:

1. Projeto Individual (80% da nota da unidade)

- Código-fonte do projeto
- Relatório técnico (PDF, 3-5 páginas)
- Link para repositório público no README

2. Listas de Exercícios (20% da nota da unidade)

- Lista da Semana 2 (Variáveis, Tipos e Operadores)
- Lista da Semana 3 (Estruturas Condicionais)
- Lista da Semana 4 - Parte A (Problemas com repetição)
- Lista da Semana 4 - Parte B (Análise de código e padrões)

3. Vídeo de Demonstração (5-8 minutos)

- Link no README principal



Estrutura Obrigatória do Arquivo

Estrutura Atualizada Recomendada:





```
sobrenome-nome-itp-u1-2025-2.zip
├── README.md                # Informações gerais e links
├── projeto/                 # Projeto principal da unidade
│   ├── src/                 # Código-fonte principal
│   │   ├── main.c           # Arquivo principal
│   │   └── [outros arquivos .c/.h]
│   ├── docs/                # Documentação do projeto
│   │   └── relatorio-u1.pdf  # Relatório técnico obrigatório
│   ├── Makefile             # Script de compilação (opcional)
│   └── README.md            # Instruções específicas do projeto
└── listas/                  # Soluções das listas
    ├── semana2-variaveis/   # Lista da Semana 2
    │   ├── problema1.c      # Calculadora de IMC
    │   └── problema2.c      # Conversão de temperatura
```

— problema3.c	# Cálculo de juros compostos
— problema4.c	# Operações aritméticas básicas
— README.md	# Instruções de compilação
— semana3-condicionais/	# Lista da Semana 3
— problema1.c	# Classificação de IMC
— problema2.c	# Calculadora de energia elétrica
— problema3.c	# Sistema de notas
— problema4.c	# Pedra, papel, tesoura
— problema5.c	# Calculadora de desconto progressivo
— problema6.c	# Diagnóstico médico simples
— problema7.c	# Sistema de equações do 2º grau
— problema8.c	# Validador de triângulos
— README.md	# Instruções de compilação
— semana4-repeticoes-a/	# Lista 4A – Problemas práticos
— problema1.c	
— problema2.c	
— problema3.c	
— problema4.c	
— README.md	
— semana4-repeticoes-b/	# Lista 4B – Análise e padrões
— questao1.txt	# Resposta teórica
— questao2.txt	# Resposta teórica
— questao3.txt	# Resposta teórica
— questao4.txt	# Resposta teórica
— questao5.c	# Código do padrão
— questao6.c	
— questao7.c	
— questao8.c	
— questao9.c	
— questao10.c	
— questao11.c	
— README.md	

Detalhamento por Componente

1. Projeto Individual (80% da nota)

1.1 Requisitos Técnicos Mínimos:

-  Implementação em linguagem C
-  Uso de **todos os conceitos da Unidade 1**:
 - Variáveis com tipos bem-definidos
 - Operações aritméticas e relacionais
 - Vetores (arrays unidimensionais)
 - Comandos condicionais (if/else)
 - Comandos de repetição (for/while/do-while)
 - Funções (pelo menos 3 funções além da main)
-  Interface de linha de comando (CLI)
-  Código original (não copiado de repositórios públicos)

-  Complexidade média ou alta

1.2 Sugestões de Projetos (caso não tenha escolhido ainda):

- Sistema de Gerenciamento de Biblioteca
- Jogo da Velha com IA Básica
- Calculadora Científica com Histórico
- Gerenciador de Tarefas Simples
- Simulador de Banco com Múltiplas Contas
- Conversor de Unidades
- Jogo de Campo Minado (versão texto)
- Sistema de Cadastro de Alunos
- Agenda de Contatos
- Calculadora de Matrizes Básica

2. Relatório Técnico (PDF, 3-5 páginas)

2.1 Estrutura Obrigatória:

Página 1 - Introdução e Contexto:

- Nome do projeto e objetivo
- Problema que o projeto resolve
- Justificativa da escolha do projeto

Páginas 2-3 - Análise Técnica:

- **Metodologia:** Ferramentas utilizadas (compilador, editor, etc.)
- **Aplicação dos Conceitos da U1:**
 - Como foram usadas as estruturas condicionais?
 - Qual a lógica das estruturas de repetição implementadas?
 - Como os vetores foram aplicados no projeto?
 - Organização e função das funções criadas?
- **Estruturas de Dados:** Explicação dos vetores e variáveis utilizadas

Páginas 4-5 - Implementação e Reflexão:

- **Dificuldades Encontradas:** Principais desafios técnicos
- **Soluções Implementadas:** Como foram superados os desafios
- **Organização do Código:** Justificativa da estrutura escolhida
- **Conclusão:** Aprendizados obtidos e possíveis melhorias

2.2 Perguntas Orientadoras (responda no relatório):

- Quais conceitos da Unidade 1 foram aplicados e onde?
- Como a organização em funções facilita a manutenção do código?
- Quais foram os principais desafios na implementação das estruturas de repetição?
- Como os vetores foram utilizados para resolver o problema proposto?
- Que melhorias poderiam ser implementadas nas próximas unidades?

3. Listas de Exercícios (20% da nota)

3.1 Lista Semana 2 - Variáveis, Tipos e Operadores

Problemas:

- **Problema 1** - Calculadora de IMC: Leia peso e altura, calcule IMC com 2 casas decimais
- **Problema 2** - Conversão de temperatura: Converta Celsius para Fahrenheit e Kelvin
- **Problema 3** - Cálculo de juros compostos: Calcule montante final com capital, taxa e tempo
- **Problema 4** - Operações aritméticas básicas: Soma, diferença, produto, divisão, resto e média

3.2 Lista Semana 3 - Estruturas Condicionais

Problemas:

- **Problema 1** - Classificação de IMC: Além de calcular, classifique conforme tabela
- **Problema 2** - Calculadora de energia elétrica: Calcule conta baseada em tipo de consumidor
- **Problema 3** - Sistema de notas: Calcule média e determine aprovação/recuperação/reprovação
- **Problema 4** - Pedra, papel, tesoura: Simule jogo entre dois jogadores
- **Problema 5** - Calculadora de desconto progressivo: Aplique descontos baseados no valor
- **Problema 6** - Diagnóstico médico simples: Sugira diagnóstico baseado em sintomas
- **Problema 7** - Sistema de equações do 2º grau: Resolva equações quadráticas
- **Problema 8** - Validador de triângulos: Verifique e classifique triângulos

3.3 Lista Semana 4A - Repetições Simples

4 problemas práticos:

1. **Problema 1 - Dobrando até não poder mais:** Dobrar folha até caber no bolso
2. **Problema 2 - Salve o homem aranha:** Calcular trajetória com distância euclidiana
3. **Problema 3 - Números colegas:** Calcular soma de divisores próprios
4. **Problema 4 - Jogo de dardos:** Calcular pontuação com base em distâncias

3.4 Lista Semana 4B - Análise e Padrões

Questões 1-4: Análise de código (arquivos .txt)

- **questao1.txt:** Resposta sobre o que é impresso (questão 1)
- **questao2.txt:** Resposta sobre o que é impresso (questão 2)
- **questao3.txt:** Resposta sobre o que é impresso (questão 3)
- **questao4.txt:** Resposta sobre o que é impresso (questão 4)

Questões 5-11: Implementação (arquivos .c)

- **questao5.c:** Padrão numérico triangular
- **questao6.c:** Trios pitagóricos
- **questao7.c:** Números primos em intervalo
- **questao8.c:** Compra de placas de alumínio
- **questao9.c:** Modificação da questão 8
- **questao10.c:** Pacote promocional

- **questao11.c**: Problema das 4 rainhas

4. Vídeo de Demonstração (5-8 minutos)

4.1 Estrutura Sugerida:

- **Minuto 1**: Apresentação pessoal e do projeto
- **Minutos 2-4**: Demonstração do projeto funcionando
- **Minutos 5-6**: Explicação do código (conceitos da U1 aplicados)
- **Minutos 7-8**: Dificuldades encontradas e aprendizados

4.2 Requisitos Técnicos:

- Duração: 5-8 minutos (vídeos fora desta faixa terão desconto na nota)
 - Qualidade de áudio clara
 - Demonstração real do código executando
 - Upload no YouTube (pode ser não listado) ou similar
 - Link incluído no README principal
-

Checklist de Verificação

Antes de Enviar:

Estrutura do Arquivo:

- ☐ Nome do arquivo: **sobrenome-nome-itp-u1-2025-2.zip**
- ☐ README.md principal presente
- ☐ Estrutura de pastas conforme especificado
- ☐ Todos os arquivos .c compilam sem erros

Projeto:

- ☐ Código usa todos os conceitos da U1
- ☐ Pelo menos 3 funções além da main
- ☐ Relatório técnico em PDF (3-5 páginas)
- ☐ Link para repositório público no README
- ☐ Projeto executa corretamente

Listas:

- ☐ Todos os 4 problemas da Semana 2 resolvidos
- ☐ Todos os 8 problemas da Semana 3 resolvidos
- ☐ Todos os 4 problemas da Lista 4A resolvidos
- ☐ Questões 1-4 da Lista 4B em arquivos .txt
- ☐ Questões 5-11 da Lista 4B em arquivos .c
- ☐ READMEs com instruções de compilação

Vídeo:

- ☐ Duração entre 5-8 minutos

- ☐ Demonstra o projeto funcionando
 - ☐ Link acessível incluído no README
-

Critérios de Avaliação

Distribuição da Nota da Unidade:

- **Projeto Individual:** 80%
 - Funcionalidade (40%): O programa executa conforme esperado
 - Aplicação dos conceitos (30%): Uso correto de todos os tópicos da U1
 - Qualidade do código (20%): Organização, nomenclatura, comentários
 - Relatório técnico (10%): Clareza e profundidade da análise
- **Listas de Exercícios:** 20%
 - Todas as listas têm peso igual
 - Correção da lógica e funcionamento do código

Detalhamento dos Critérios:

Funcionalidade (40% do projeto):

- Programa compila sem erros ou warnings
- Todas as funcionalidades principais funcionam
- Interface é intuitiva e amigável
- Tratamento básico de entradas inválidas

Aplicação dos Conceitos (30% do projeto):

- Uso correto de condicionais em situações apropriadas
- Implementação adequada de estruturas de repetição
- Utilização de vetores para resolver problemas
- Organização em funções com responsabilidades claras
- Declaração e uso correto de variáveis

Qualidade do Código (20% do projeto):

- Nomenclatura descritiva para variáveis e funções
- Indentação e organização visual consistente
- Comentários em partes complexas do código
- Código limpo e legível

Relatório Técnico (10% do projeto):

- Explicação clara dos conceitos aplicados
 - Reflexão sobre dificuldades e soluções
 - Análise da organização do código
 - Resposta às perguntas orientadoras
-

Instruções de Compilação

Para cada arquivo .c das listas:

```
gcc -o nome_programa nome_programa.c  
./nome_programa
```

Para o projeto (se usar Makefile):

```
make  
./projeto
```

Para o projeto (compilação manual):

```
gcc -o projeto src/main.c src/outros_arquivos.c  
./projeto
```

Importante: Todos os códigos devem compilar e executar corretamente no ambiente Linux com GCC.

⚠ Observações Importantes

Políticas de Entrega:

1. **Apenas 1 arquivo será aceito:** O SIGAA permite somente um anexo
2. **Formato obrigatório:** .zip seguindo a estrutura especificada
3. **Prazo rígido:** 30/09/2025 às 23:59 (após este horário, não será aceito)
4. **Commit considerado:** Apenas o commit mais recente até o prazo será avaliado

Originalidade:

- Código deve ser original e individual
- Não copie soluções de repositórios públicos ou colegas
- Consultas a documentação e tutoriais são permitidas
- Em caso de plágio, a nota será zero

Funcionamento:

- Todo código deve compilar sem erros
- Programas devem executar corretamente
- Dados de entrada inválidos devem ser tratados
- Interface deve ser clara e intuitiva

Dúvidas:

- **Atendimento:** Segundas e sextas durante as aulas (16:40-18:20)
- **E-mail:** Apenas para questões administrativas urgentes
- **SIGAA:** Fórum da disciplina para dúvidas técnicas

Recursos de Apoio

Materiais Disponíveis:

- Material de apoio das Semanas 2, 3 e 4
- Slides das aulas gravadas
- Exemplos de código discutidos em aula
- Lista de projetos sugeridos no README da disciplina

Ferramentas Recomendadas:

- **Compilador:** GCC (Linux/Windows via MinGW/WSL)
- **Editor:** VS Code, Code::Blocks, ou similar
- **Versionamento:** Git + GitHub/GitLab/BitBucket
- **Gravação:** OBS Studio, QuickTime, ou similar



Modelo de README Principal

```
# Introdução a Técnicas de Programação – Unidade 1
**Aluno**: [Seu Nome Completo]
**Matrícula**: [Sua Matrícula]
**Período**: 2025.2

## 📁 Estrutura do Projeto

- `projeto/`: Projeto principal da unidade
- `listas/`: Soluções das listas de exercícios
- `README.md`: Este arquivo

## 🚀 Projeto: [Nome do Seu Projeto]

**Descrição**: [Breve descrição do que o projeto faz]

**Repositório**: [Link para repositório público]

**Vídeo de Demonstração**: [Link para o vídeo]

### Funcionalidades Implementadas:
- [Lista das principais funcionalidades]

### Conceitos da U1 Aplicados:
- Estruturas condicionais: [onde foram usadas]
- Estruturas de repetição: [onde foram usadas]
- Vetores: [como foram aplicados]
- Funções: [quantas e suas responsabilidades]
```


📖 Listas de Exercícios

Semana 2 – Variáveis, Tipos e Operadores:

- ☒ Problema 1: Calculadora de IMC
- ☒ Problema 2: Conversão de temperatura
- ☒ Problema 3: Cálculo de juros compostos
- ☒ Problema 4: Operações aritméticas básicas

Semana 3 – Condicionais:

- ☒ Problema 1: Classificação de IMC
- ☒ Problema 2: Calculadora de energia elétrica
- ☒ Problema 3: Sistema de notas
- ☒ Problema 4: Pedra, papel, tesoura
- ☒ Problema 5: Calculadora de desconto progressivo
- ☒ Problema 6: Diagnóstico médico simples
- ☒ Problema 7: Sistema de equações do 2º grau
- ☒ Problema 8: Validador de triângulos

Semana 4A – Repetições:

- ☒ Problema 1: Dobrar folha
- ☒ Problema 2: Homem Aranha
- ☒ Problema 3: Números colegas
- ☒ Problema 4: Jogo de dardos

Semana 4B – Análise e Padrões:

- ☒ Questões 1–4: Análise de código
- ☒ Questões 5–11: Implementações

🎯 Principais Aprendizados

[Reflita sobre os principais conceitos aprendidos na U1]

🛠 Ambiente de Desenvolvimento

- ****SO****: [Windows/Linux/macOS]
- ****Compilador****: GCC versão [X.X]
- ****Editor****: [Nome do editor usado]