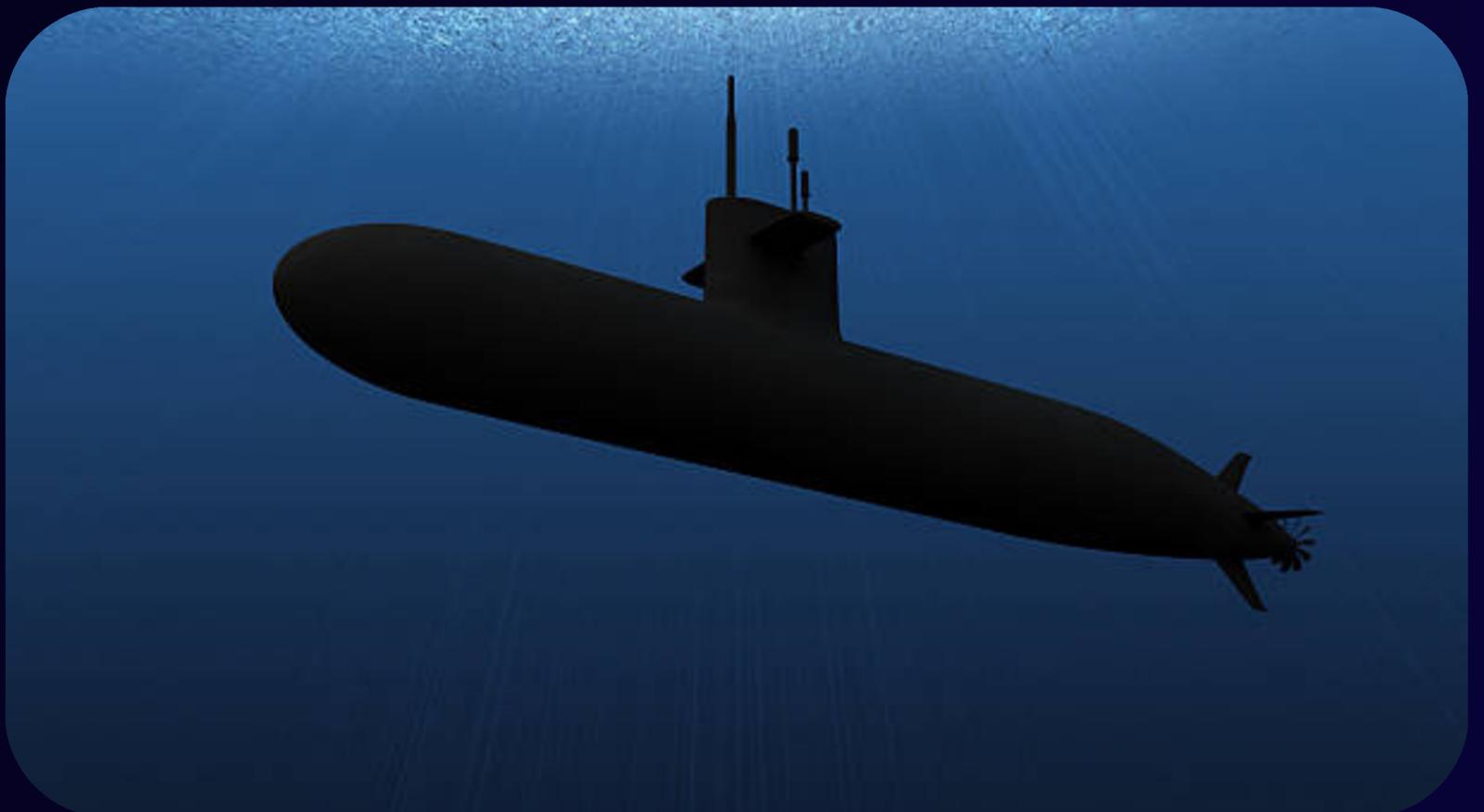


APS - Submarinos

LÓGICA DA COMPUTAÇÃO - 2025.1

Por que da linguagem?

- **Contextualizar** ensino de compiladores com um tema lúdico: submarinos.
- **Complementar** os aprendizados de análise léxica, sintática e geração de código.
- **Integrar** conceitos reais de navegação simulada em comandos específicos. (Ou tentar...)
- Tornar a linguagem divertida, mas tecnicamente sólida.



Estrutura da Linguagem

"Estruturas básicas de uma linguagem de programação:
variáveis, condicionais e loops."

- Variáveis (declaração e atribuição)
- Comandos (se, enquanto, dizer, etc.)
- Estrutura EBNF formal
- Compilação para LLVM IR

Ferramentas usadas:

- Flex (análise léxica)
- Bison (análise sintática)
- LLVM IR (código intermediário executável com lli)



Funcionalidades Implementadas

Comandos da linguagem:

- var x : int;
- x = 10;
- se x < 5 { ... } senao { ... }
- enquanto x < 10 { ... }
- dizer("texto");, dizer(x);
- routine nome { ... }, call nome;
(suporte parcial)

Comandos específicos de submarino:

- subir;, descer;, inclinar(45);
- navegar_para(10, 5);
- acelerar(2);, parar;, status;

Todos traduzidos em @printf em LLVM IR.

Numa possível V2 podemos aproveitar o suporte parcial de chamada de rotina para implementar essas funções de fato!

Exemplo de Código 1: Condicional

```
inicio() {  
    var destino: int;  
    destino = 7;  
    dizer("Preparando para navegação");  
    se destino < 10 {  
        dizer("Destino curto. Acelerando...");  
        acelerar(2);  
        navegar_para(destino, 3);  
    } senao {  
        dizer("Destino longo. Aguardando...");  
    }  
}
```

Output:

Preparando para navegação
Destino curto.
Acelerando...
[NAV] acelerando: 2
[NAV] indo para (7,3)

Exemplo de Código 2: Laço While

```
inicio() {  
    var distacia: int;  
    distacia = 0;  
    dizer("Preparando para navegação");  
    enquanto distacia < 3 {  
        dizer("Navegando");  
        distacia = distacia + 1;  
    };  
}
```

Output:

Navegando
Navegando
Navegando

Exemplo de Código 3: Sequência de comandos marítimos

```
inicio() {  
    dizer("Subindo...");  
    subir;  
    dizer("Descendo...");  
    descer;  
    dizer("Inclinando para manobra");  
    inclinar(45);  
    dizer("Parando...");  
    parar;  
}
```

Output:

Subindo...
Descendo...
Inclinando para manobra
Parando...

Testes e Execução

Após criação de scripts .sub, compilamos com:

```
flex lexer.l  
bison -d parser.y  
gcc -o submarino parser.tab.c lex.yy.c -lfl  
.submarino < test.sub  
lli output.ll
```

vitorbandeira1/
APS_LogComp25-1



1 Contributor 0 Issues 0 Stars 0 Forks

vitorbandeira1/APS_LogComp25-1
Contribute to vitorbandeira1/APS_LogComp25-1 development by creating an account on GitHub.

[GitHub](#)

Relato/Curiosidade:

Trabalhar com LLVM IR como compilador da minha linguagem foi extremamente desafiador, mesmo com o auxílio de IAs. Foi interessante entender melhor como essa ferramenta funciona e seu potencial para otimização e estabilidade, graças às regras bem definidas. Começar a enxergar a “caixa-preta” dos compiladores deixou evidente a complexidade por trás de cada linha de código, suas chamadas de rotina e declarações

Obrigado!