Técnicas para Programação Competitiva

Plano de Ensino e Introdução

Samuel da Silva Feitosa

Aula 1

Apresentação do Professor

- Formação
  - Ciência da Computação / Licenciatura
  - Especializações em Gestão de Empresas, Educação Profissional e Tecnológica e Gestão da Educação Profissional e Tecnológica
  - Mestrado e Doutorado em Computação
  - Período do Doutorado na Universidade de Utrecht na Holanda
- Experiência
  - Professor desde 2011 - Unochapecó (2011-2013), UFSM (2014), IFSC (2016-2021), UFFS (2021-atual)
  - Analista e Desenvolvedor de Sistemas
  - Implantação de Software / Servidores Linux
  - Programação em Haskell, Agda, Java, C/C++, Python, dentre outras

Apresentação do Estudante

- Nome
- Idade
- Trabalha? Na área? Estágio ou emprego?
- Já participou da Maratona de Programação?
- Já resolveu problemas em plataformas online?

Plano de Ensino

Ementa

- Técnicas avançadas de programação.
- Programação competitiva.
- Estruturas de dados avançadas.
- Algoritmos gulosos.
- Algoritmos de divisão e conquista.
- Programação dinâmica.
- Problemas combinatoriais.
- Análise de algoritmos.
- Otimização de código.

Objetivos

- Geral
  - Estudar técnicas avançadas de programação para a resolução de problemas complexos, praticando a implementação das técnicas em diversos exercícios e comparando analiticamente o resultado com os algoritmos de força bruta. Aprimorar a criatividade e as habilidades necessárias para competições de programação.
- Específicos
  - Aprimorar a criatividade e as habilidades necessárias para competições de programação.
  - Fomentar o espírito de trabalho em equipe e o interesse por Programação Competitiva.
  - Motivar a formação de times qualificados para a Maratona de Programação.
  - Desenvolver as habilidades acadêmicas de exposição de resultados e argumentação.

Organização

- A disciplina será organizada da seguinte forma:
  - Aulas em laboratório de informática (Sala 409-B).
  - Algumas aulas com apresentação de conceitos importantes para Programação Competitiva.
  - Outras aulas para resolução de problemas, simulação de competições, apresentação de soluções, etc.

Procedimentos Metodológicos

- Aulas expositivas-dialogadas para apresentação de conceitos.
- Aprendizagem baseada em problemas (PBL).
- Apresentação de soluções pelos estudantes.
- Programação pareada.
- Competições.

Avaliação

- NP1 - Participação, assiduidade e entrega de atividades desenvolvidas em sala e extraclasse (20%).
- NP2 - Resolução de listas de exercícios (plataformas) (60%).
- NP3 - Apresentação de soluções implementadas (20%)

Média Final: NP1 * 0,2 + NP2 * 0,6 + NP3 * 0,2

Referências Básicas

LAAKSONEN, ANTTI. **Guide to Competitive Programming:** Learning and Improving Algorithms Through Contests, 2ª Edição. Springer, 2020.

HALIM, S.; HALIM, F. **Competitive Programming 3:** The New Lower Bound of Programming Contests, Lulu, 2013.

SKIENA, S. S.; REVILLA, M. **Programming Challenges**, 1ª edição. Springer, 2003.

Observações do Professor

- Evitar entradas e saídas da sala de aula.
- Atentar para os prazos de entregas parciais.
- Atendimento nas segundas-feiras das 16:30 até as 17:30, ou mediante agendamento.
- Plágio em atividades avaliativas não será tolerado.
  - Todos os envolvidos na atividade terão nota zero atribuída.
- Comunicados referentes a disciplina serão dados em sala de aula.
  - É responsabilidade do estudante buscar informações caso falte alguma aula.

Avaliação Diagnóstica

- Considerando os tópicos descritos na **ementa** dessa disciplina, escreva um texto contendo o seguinte:
  - Uma descrição dos seus gostos (como estudante, profissionais e pessoais) e de como você leva a vida na cidade onde mora.
  - Uma descrição dos assuntos presentes na ementa que você já estudou, explicitando se este estudo ocorreu em outras disciplinas ou por conta própria. Se preciso faça uma breve pesquisa na internet usando o seu smartphone.
  - Se já participou de alguma competição de Prog. Competitiva e se já resolveu problemas nas plataformas de julgamento online (quais?).
- Seja o mais detalhista possível, pois estas informações serão úteis para conhecer vocês e como organizar os conteúdos e atividades da disciplina.

Introdução à Programação Competitiva

O que é Programação Competitiva?

- Programação competitiva combina dois tópicos:
  - **Projeto de Algoritmos**
    - Projetar algoritmos eficientes que resolvem problemas computacionais.
    - Requer habilidades matemáticas e de resolução de problemas.
    - Geralmente a solução é uma combinação de métodos bem conhecidos.
  - **Implementação de Algoritmos**
    - Soluções são avaliadas e testadas usando um conjunto de casos de teste.
    - Programas geralmente são pequenos, e devem ser escritos rapidamente.

Linguagens e Competições

- Vamos adotar a linguagem C++ para esta disciplina
  - É utilizada por mais de 70% dos participantes de eventos de Programação Competitiva.
- Principais competições:
  - **ICPC - Maratona de Programação** (regional, nacional, mundial), para alunos de graduação.
  - **IOI - Olimpíada de Informática**, para alunos do ensino médio.
  - **Competições Online**, tendo o CodeForces.com como o mais ativo atualmente, com competições semanais.

Dicas para a Disciplina

- Aprender programação competitiva requer bastante trabalho.
  - Quanto mais se pratica, mais fácil será de resolver novos problemas.
- Qualidade do problema é mais importante do que quantidade.
  - Problemas mais elaborados e mais complexos devem elevar suas habilidades.
- A maioria dos problemas podem ser resolvidos usando algoritmos pequenos e simples.
  - Abordagem de problemas difíceis usando ferramentas simples.
- Interpretação de problemas em Inglês.
  - A maioria das plataformas apresentam a descrição dos problemas em Inglês. Acostumem-se com isso…

CSES Problem Set

- Vamos resolver muitos problemas disponíveis na plataforma CSES.
  - https://cses.fi/problemset/
  - Os problemas são organizados por ordem de dificuldade.
  - O livro que consta na primeira referência do plano de ensino, trata dos problemas dessa lista.
- Vamos ver como resolver o primeiro problema da lista, chamado *Weird Algorithm*.

Weird Algorithm - Descrição

Consider an algorithm that takes as input a positive integer n. If n is even, the algorithm divides it by two, and if n is odd, the algorithm multiplies it by three and adds one. The algorithm repeats this, until n is one. For example, the sequence for n = 3 is as follows:

$$3 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$$

Your task is to simulate the execution of the algorithm for a given value of n.

Weird Algorithm - Entrada e Saída

Input

The only input line contains an integer n.

Output

Print a line that contains all values of n during the algorithm.

Constraints

- $1 \leq n \leq 10^6$

Example

Input:

3

Output:

3 10 5 16 8 4 2 1

Weird Algorithm - Solução (1)

```
#include <iostream>

using namespace std;

int main() {
    int n;
    cin >> n;
    while (true) {
        cout << n << " ";
        if (n == 1) break;
        if (n%2 == 0) n /= 2;
        else n = n*3+1;
    }
    cout << "\n";
}
```

- Resultado dos testes:

Test	Verdict	Time (s)
#1	ACCEPTED	0.06 / 1.00
#2	ACCEPTED	0.06 / 1.00
#3	ACCEPTED	0.07 / 1.00
#4	ACCEPTED	0.06 / 1.00
#5	ACCEPTED	0.06 / 1.00
#6	TIME LIMIT EXCEEDED	– / 1.00
#7	TIME LIMIT EXCEEDED	– / 1.00
#8	WRONG ANSWER	0.07 / 1.00
#9	TIME LIMIT EXCEEDED	– / 1.00
#10	ACCEPTED	0.06 / 1.00

Weird Algorithm - Solução (2)

- Um dos casos de erro apresentados tem n = 138367.
  - Se rodarmos o código localmente, veremos que ele é de fato lento.
  - Na verdade, este código nunca termina.

Weird Algorithm - Solução (2)

- Um dos casos de erro apresentados tem n = 138367.
  - Se rodarmos o código localmente, veremos que ele é de fato lento.
  - Na verdade, este código nunca termina.
- O motivo para este código falhar é que *n* pode crescer muito durante a simulação, estourando o limite de uma variável inteira.
  - Para resolver o problema, basta utilizar o tipo *long long* para a variável *n*.
- Como este exemplo mostra, mesmo algoritmos simples podem conter bugs.

Considerações Finais

- Discussão sobre a forma que a disciplina será conduzida.
  - Organização, procedimentos metodológicos e avaliação.
- Realizamos a avaliação diagnóstica da turma.
- Discutimos sobre programação competitiva.
- Realizamos um primeiro exercício utilizando a plataforma CSES.