Estratégias de Treinamento

Prof. Edson Alves

2018

Faculdade UnB Gama

Sumário

Recursos para prática de programação competitiva

Codeforces

- Codeforces (`codeforces.com`) é um juiz online que hospeda competições regularmente
- É, atualmente, o melhor juiz online para treinamento de alto desempenho
- Tem como vantagem a transparência: as soluções dos problemas de todos os participantes ficam em aberto para estudo, e também são publicados editoriais com comentários sobre as soluções dos problemas
- Os contests podem ser feitos posteriormente, através da opção de participação virtual
- Vale a pena fazer ao menos os contests educacionais virtualmente

UVA Online Judge

- O UVA (*uva.onlinejudge.org*) é um juiz online com uma imensa base de problemas
- Já foi a principal plataforma de treino, mas tem sido abandonado em favor do Codeforces
- Contudo, os problemas do UVA são mais próximos dos problemas do ACM ICPC, de forma que um participante de alto nível deve ter contato com estes problemas também
- A plataforma uHunt (`uhunt.onlinejudge.org`) sistematiza a apresentação dos problemas do UVA, e lista também o progresso do usuário nos problemas listados no CP3
- O uHunt também permite a criação de contests virtuais
- O Live Archive (`https://icpcarchive.ecs.baylor.edu/`) traz os problemas da maioria dos eventos do ACM ICPC

Livros Didáticos

- O livro *Competitive Programming 3*, dos irmãos Halim, foi a grande referência de programação competitiva da época
- Ele traz uma série de exercícios sugeridos do UVA, que estão listados no uHunt
- O livro *Competitive Programmer's Handbook*, de Antti Laaksonen, é um livro mais recente (2018), e que traz estruturas e algoritmos ausentes no CP3
- Além disso, o PDF com o livro é gratuito
- Contudo, ele não dá implementação completas dos algoritmos, deixando-as a cargo dos leitores

Categorização de problemas

Categorias de problemas

- Os eventos ACM ICPC tem, em geral, de 8 a 12 problemas, a serem resolvidos em 5 horas
- Os *rounds* do Codeforces tem entre 4 e 7 problemas, a serem resolvidos em 2 horas
- Os problemas podem ser classificados em 10 Categorias
  1. `Ad-hoc`
  2. `Busca Completa`
  3. `Dividir e Conquistar`
  4. `Gulosos`
  5. `Programação Dinâmica`
  6. `Grafos`
  7. `Matemática`
  8. `Strings`
  9. `Geometria Computacional`
  10. `Estruturas de Dados`
- Há problemas que não se enquadram nestas categorias, porém com uma frequência incomum de aparição
- As subcategorias que um competidor deve dominar são: ordenação, recursão e a própria linguagem de programação escolhida

Classificação pessoal dos problemas

- Um competidor pode classificar os problemas em 3 cateogorias
  A. `Já resolvi um parecido e posso resolver de novo rapidamente`
  B. `Já vi um parecido e sei que não consigo resolver`
  C. `Nunca vi`
- Para se tornar um competidor efetivo é preciso
  1. classificar a maioria dos problemas como `A`
  2. minimizar os problemas `B`
  3. usar os conhecimentos acumulados no treinamento para atacar os problemas `C`
- É importante manter um registro dos problemas do tipo `B` e `C` encontrados, e tentar resolvê-los posteriormente, através da discussão com os outros competidores ou leitura dos editoriais e soluções disponíveis
- A resolução de problemas que antes eram inacessíveis promove o crescimento técnico e profissional do competidor

Teste do código da solução

- Os exemplos de entrada e saída do problema, em geral, são triviais
- No caso de uma solução `WA`, é preciso achar um teste que quebre a solução
- Para gerar estes testes, é preciso avaliar os *corner cases*:
  1. os maiores valores possíveis para as entradas
  2. os menores valores possíveis para as entradas
  3. variáveis com valores zerados ou negativos
  4. grafos desconetados, polígonos não convexos, polígonos degenerados
  5. todos as entradas iguais
  6. entradas em ordem crescente ou decrescente
- Também é bom verificar se a solução proposta está assumindo alguma característica da entrada que não foi descrita explicitamente no problema (ordenação, valores máximos, etc)

Trabalho em equipe

Referências

1. **HALIM**, Felix; **HALIM**, Steve. *Competitive Programming 3*, 2010.
2. **LAAKSONEN**, Antti. *Competitive Programmer's Handbook*, 2018.
3. **SKIENA**, Steven S.; **REVILLA**, Miguel A. *Programming Challenges*, 2003.
4. CppReference[1].

[1] https://en.cppreference.com/w/