

Programação e Desenvolvimento de Software I

Max Santiago

max.santiago@dcc.ufmg.br

DCC - Departamento de Ciência da Computação

Objetivos

- Introduzir o aluno aos conceitos de **algoritmos e estruturas de dados**.

Objetivos e metas

- Introduzir o aluno aos conceitos de **algoritmos e estruturas de dados**
 - Noções da organização e funcionamento de um computador.
 - Noções de linguagens imperativas (vamos utilizar a linguagem C).
 - Noções de estruturas de dados.

Por que aprender programação?

- Diversas áreas se beneficiam de programação.
 - Simulações, análise de dados, sistemas embutidos, sensores.
 - Matemática computacional.
 - Física computacional.
 - Química, Estatística, Economia, Linguística...

Por que aprender programação?

- Será necessário para outras disciplinas
 - PDS II.
 - Circuitos digitais (pode ser).
 - Análise numérica.
 - Redes de computadores.
 - Informática industrial.
 - Controle digital.
 - Pesquisa operacional.
 - Optativas: bancos de dados, redes de computadores, visão computacional, ...

Avaliações

- 3 provas (teóricas e individuais)
 - 1a. 25 pontos
 - 2a. 25 pontos
 - 3a. 25 pontos
- Laboratório 10 pontos
- Trabalho prático 15 pontos (individual)

O nosso curso

- Aulas práticas:
 - Toda terça-feira.
 - Resolução de exercícios e dúvidas.
 - Os exercícios serão disponibilizados no início da aula.
 - Entrega dos exercícios até terça (próxima aula).
 - As soluções serão liberadas toda quarta.
- Aulas teóricas.
 - Apresentação dos conceitos.
 - Toda quinta-feira.

Aulas práticas

- Aulas no laboratório (LAICO B.307) / (Sala 306).
- Aulas práticas com **avaliação**:
 - Serão previamente avisados.
 - Entrega, até o final da aula, da solução de um problema simples de programação.
 - Valor de cada entrega: 1 ponto.

- Conteúdo base: livro, slides e exercícios.
- Dou dicas sobre a prova durante as aulas.
 - Comentários em sala de aula.
 - Exercícios parecidos.
- A resposta pode não estar no material, mas a matéria lecionada é a base teórica para resolvê-la.

Provas

- Presenciais.
- Duração: 1 hora e 40 minutos.
- No papel no VPL do Moodle (a definir)
 - Ambiente virtual de programação com correção automática.

Revisão da correção das provas

- Até duas semanas depois da entrega da nota da avaliação
 - Frequência é fundamental !
 - Prestem atenção, estudem !
 - Importante revisar, pois a correção automática pode levar a erros injustos (ex: na sexta casa decimal)

Práticas regulares

- Entrega no Moodle: arquivos .c e .h
- Pontuação do tipo “entregou, levou”.
- Faremos as correções de todos os exercícios práticos.

Práticas avaliativas

- Avaliativas (treino para a prova).
- Correção automática no VPL do Moodle.
- Pontuação depende da correção das questões.
- Mesmo formato das provas .

Trabalho prático

- Momento para realizar um projeto mais longo e complexo
- Julgamento do código
 - Comentários, facilidade de leitura, indentação
- Julgamento da documentação
 - Estruturação, clareza e coesão, conteúdo

Algumas ideias sobre o trabalho

“É uma excelente oportunidade para desenvolver a capacidade de aprender novas ferramentas.”



Algumas ideias sobre o trabalho

“Permite que o aluno se divirta desenvolvendo um trabalho prático.”



Algumas ideias sobre o trabalho

“Permite o desenvolvimento de um trabalho maior, mais complexo, além de incentivar a criatividade do aluno.”



Algumas ideias sobre o trabalho

“TP totalmente não condizente com a realidade, fruto de um devaneio de um profissional frustrado.”



Avaliação do trabalho prático

- Nota final =
(
até 20 pontos para o trabalho básico considerando os itens
mencionados anteriormente
+ pontos extras
) * nota da prova oral

Avaliação do trabalho prático

- Prova oral consistirá de X (ex: 3) perguntas diretas sobre o código
- Nota da arguição = número de perguntas respondidas corretamente em tempo hábil dividido por X
 - Ex: se $x = 3$, valores possíveis: 1, $2/3$, $1/3$, 0
- Se você sabe o que fez no TP, vai tirar 1 na prova oral!
- No entanto, vários alunos já tiraram 0 e $1/3$ na prova oral :~(

Avaliação do trabalho prático

- Pontos extras só serão dados aos alunos que ficarem com mais de 50% dos pontos **nas provas** (mais de 37,5 pontos)
 - 37 não é > 50%, **inegociável**.
- Pontos extras SÓ servem para aumentar o conceito de D para C, C para B e B para A
- Ou seja, não espere passar com os pontos extras (E para D)!

tec

'Flappy Bird' chega a 50 milhões de downloads e rende US\$ 50 mil por dia

DE SÃO PAULO

08/02/2014 12h10

[Recomendar](#) 178 [Tweetar](#) 51 [G+1](#) 4 [OUVR O TEXTO](#) [Mais opções](#)

Com uma dificuldade absurda disfarçada em gráficos simples, o jogo "Flappy Bird" virou fenômeno.

Primeiro em diversas categorias na App Store e no Google Play, ele já foi baixado 50 milhões de vezes e acumula mais de 47 mil avaliações.

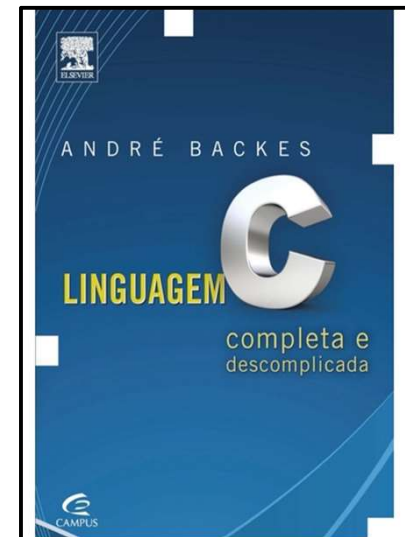
Dong Nguyen, o vietnamita que criou o jogo, diz faturar US\$ 50 mil por dia com publicidade.

O desenvolvedor é independente e atingiu o sucesso sem a ajuda de um grande estúdio. ★ ★ ★



Bibliografia

- Livro-texto:
 - Linguagem C completa e descomplicada, André Backes
- Outros:
 - Introdução às Estruturas de Dados, Waldemar Celes
 - Projeto de Algoritmos com implementação em PASCAL e C, 3a edição, Nivio Ziviani
 - Algoritmos estruturados, 3a edição, Harry Farrer, Becker, Faria, Matos, dos Santos, Maia



Linguagem C

- Criada em 1972
 - 51 anos de idade
 - Para computação → bem velho
- Existem linguagens mais recentes? Sim.

- Existem linguagens mais recentes? Sim
 - A escolha de uma linguagem para aprender a programar é um problema complicado
 - C serve para entender o funcionamento do computador melhor
 - Baixo nível
 - Com o tempo alunos podem usar o conhecimento para aprender outras linguagens
 - Depois de bastante prática, chavear linguagens é algo simples
 - Plano didático da UFMG é quase todo em C

Notas e frequência

- Não reprovado por frequência
 - SE o aluno tiver aproveitamento superior a 60%
- Se for infrequente (frequência < 75%):
 - Não ajuda a mudar de conceito
 - Não ajuda a passar (mesmo que seja por 1 ponto)
 - Não tem direito a exame especial
 - Infrequente e nota menor que 60: conceito F

Notas e frequência

- Listas de presença em todas as aulas
- Não precisa vir à aula
 - Se quiser ficar conversando com os colegas

É só tirar 60 pontos ou mais...

Notas e frequência

- As aulas são importantes:
 - Posso dar dicas sobre questões de prova
 - Posso propor um exercício parecido ao da prova
 - Posso mencionar algo que não está no livro, mas cai na prova

Exame especial

- Especial, como diz o nome
 - Difícil: matéria do semestre inteiro
 - Muitas questões: preciso avaliar a matéria como um todo
- Sugestão: evitem fazer o exame especial
 - É mais fácil passar com as provas e trabalhos
 - Férias começam mais cedo

Extra-classe

- Teremos monitores
- Estou disponível fora do horário de aula
 - Marcando horário
 - Por e-mail: melhor para notas/correções
- Fórum do Moodle/Minha UFMG: dúvidas sobre a matéria/programação

É fácil aprender programação?

Teach Yourself Programming in Ten Years

Peter Norvig

Why is everyone in such a rush?

Walk into any bookstore, and you'll see how to *Teach Yourself Java in 24 Hours* alongside endless variations offering to teach C, SQL, Ruby, Algorithms, and so on in a few days or hours. The Amazon advanced search for [\[title: teach, yourself, hours, since: 2000\]](#) and found 512 such books. Of the top ten, nine are programming books (the other is about bookkeeping). Similar results come from replacing "teach yourself" with "learn" or "hours" with "days."

The conclusion is that either people are in a big rush to learn about programming, or that programming is somehow fabulously easier to learn than anything else. Felleisen *et al.* give a nod to this trend in their book [How to Design Programs](#), when they say "Bad programming is easy. *Idiots* can learn it in 21 days, even if they are *dummies*." The Abtruse Goose comic also had [their take](#).

Translations

Thanks to the following authors, translations of this page are available in:

[Arabic](#)
([Mohamed A. Yahya](#))

العربية

[Bulgarian](#)
([Boyko Bantchev](#))

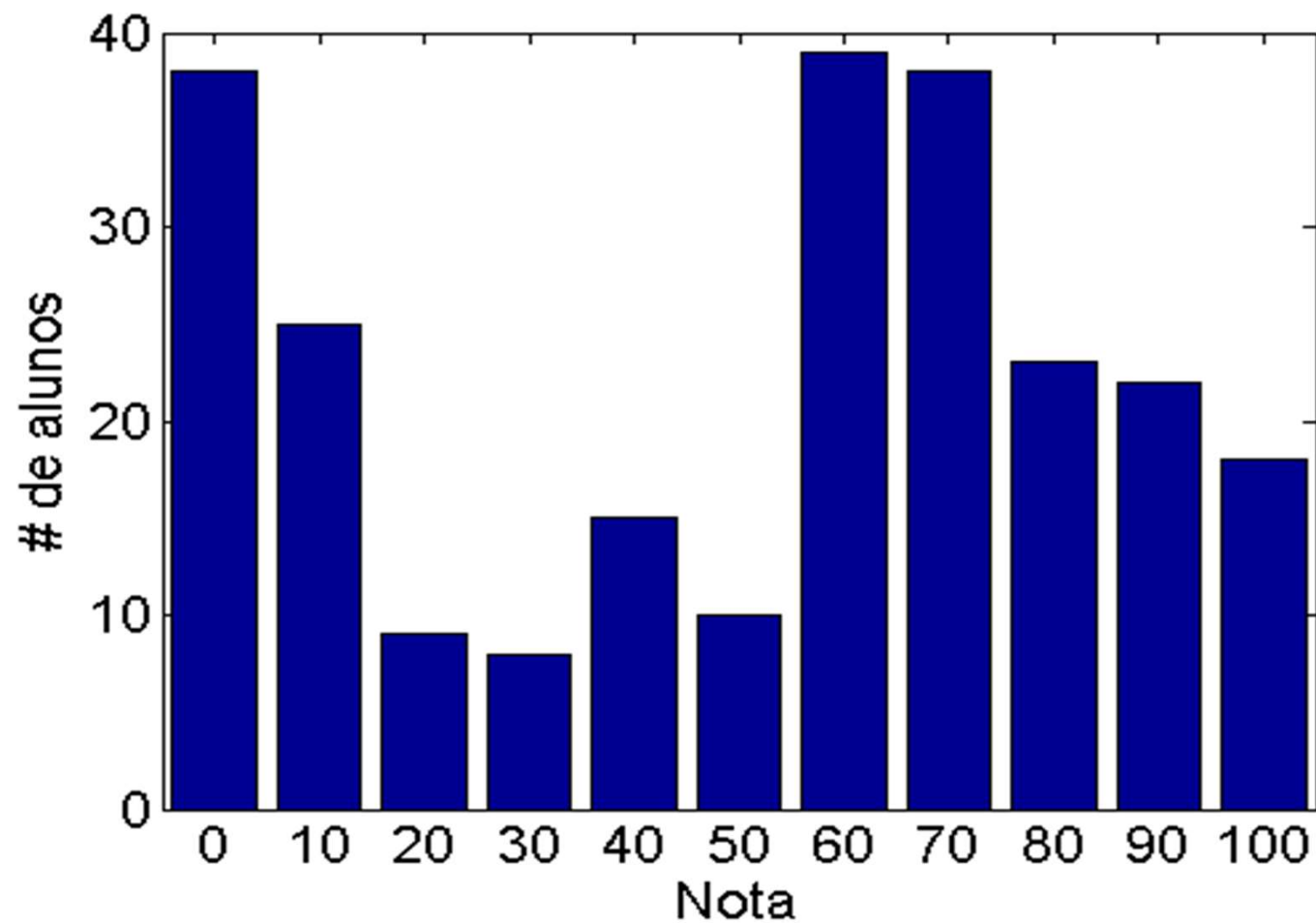
Observações

- O curso não é fácil
- Diferente do segundo grau: objetivo é formar analistas, não recitadores
- Programação é difícil de aprender
 - Diferente de tudo o que vocês já aprenderam antes
 - Matemática + lógica + “Arte”

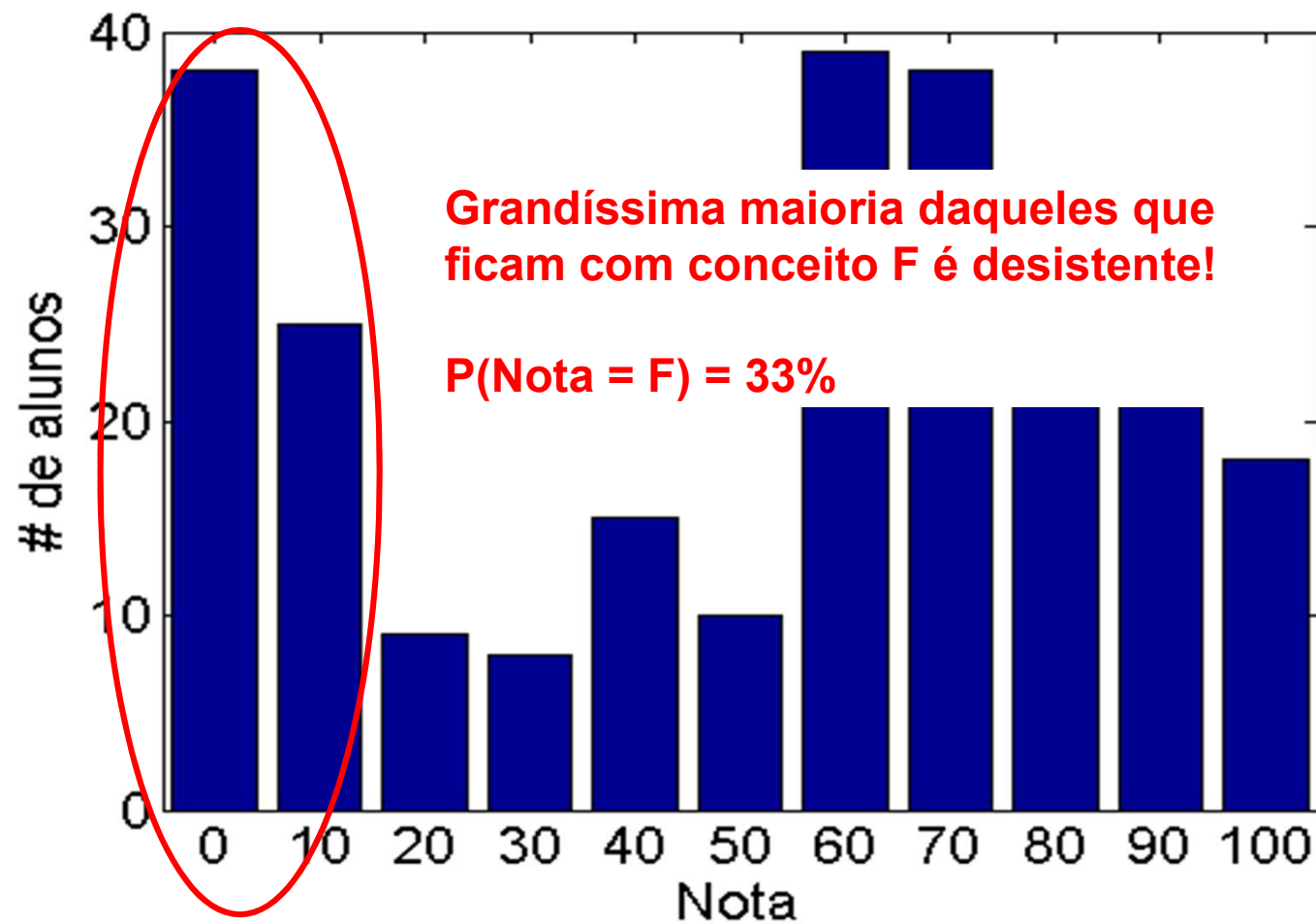
Observações

- Dedicação é fundamental
 - **Só se aprende a programar programando**
 - Trabalhos práticos requerem afinho e muitas horas de esforço
 - Provas e trabalhos exigindo conceitos + raciocínio lógico + análise de problemas

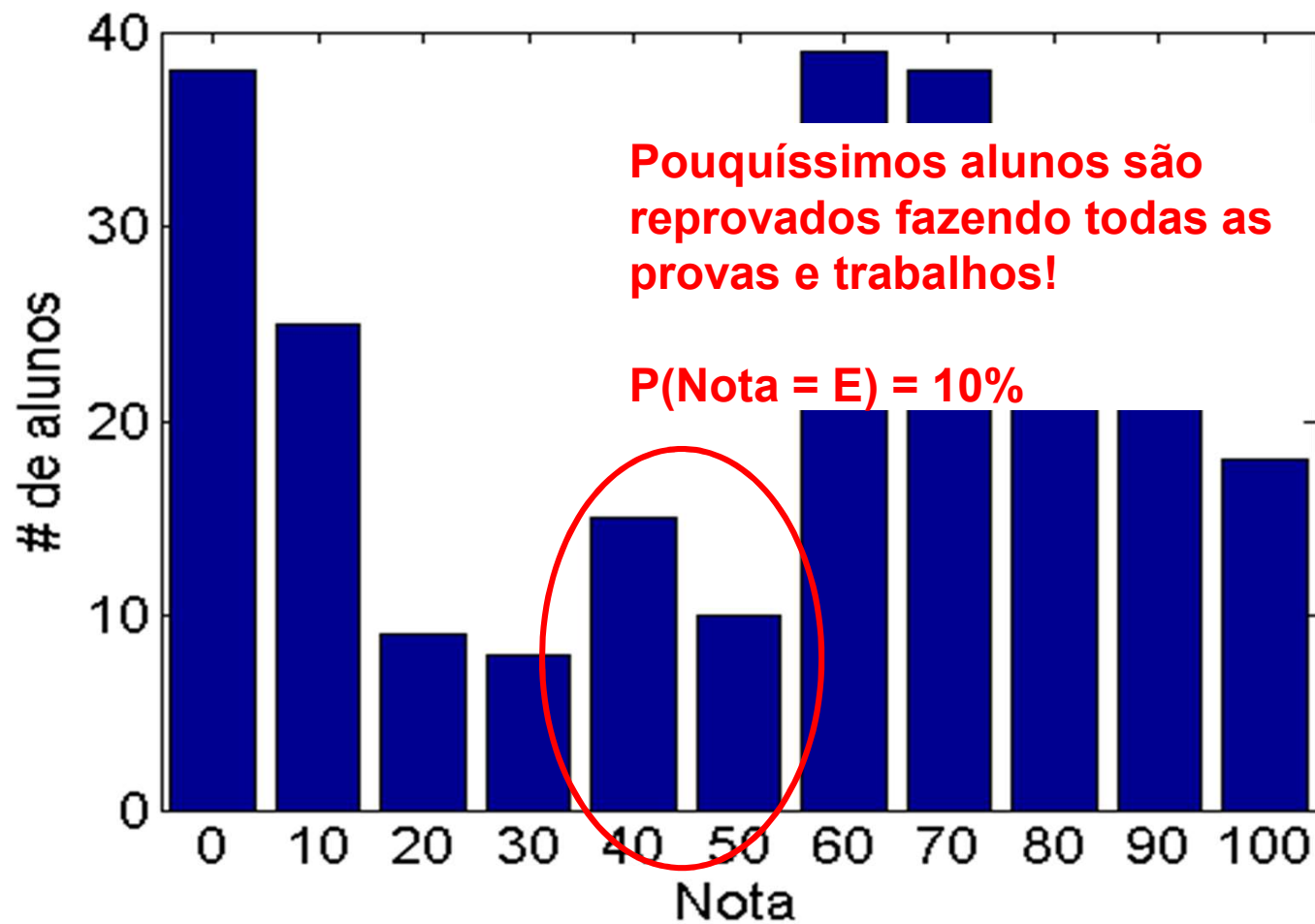
Desempenho dos Alunos



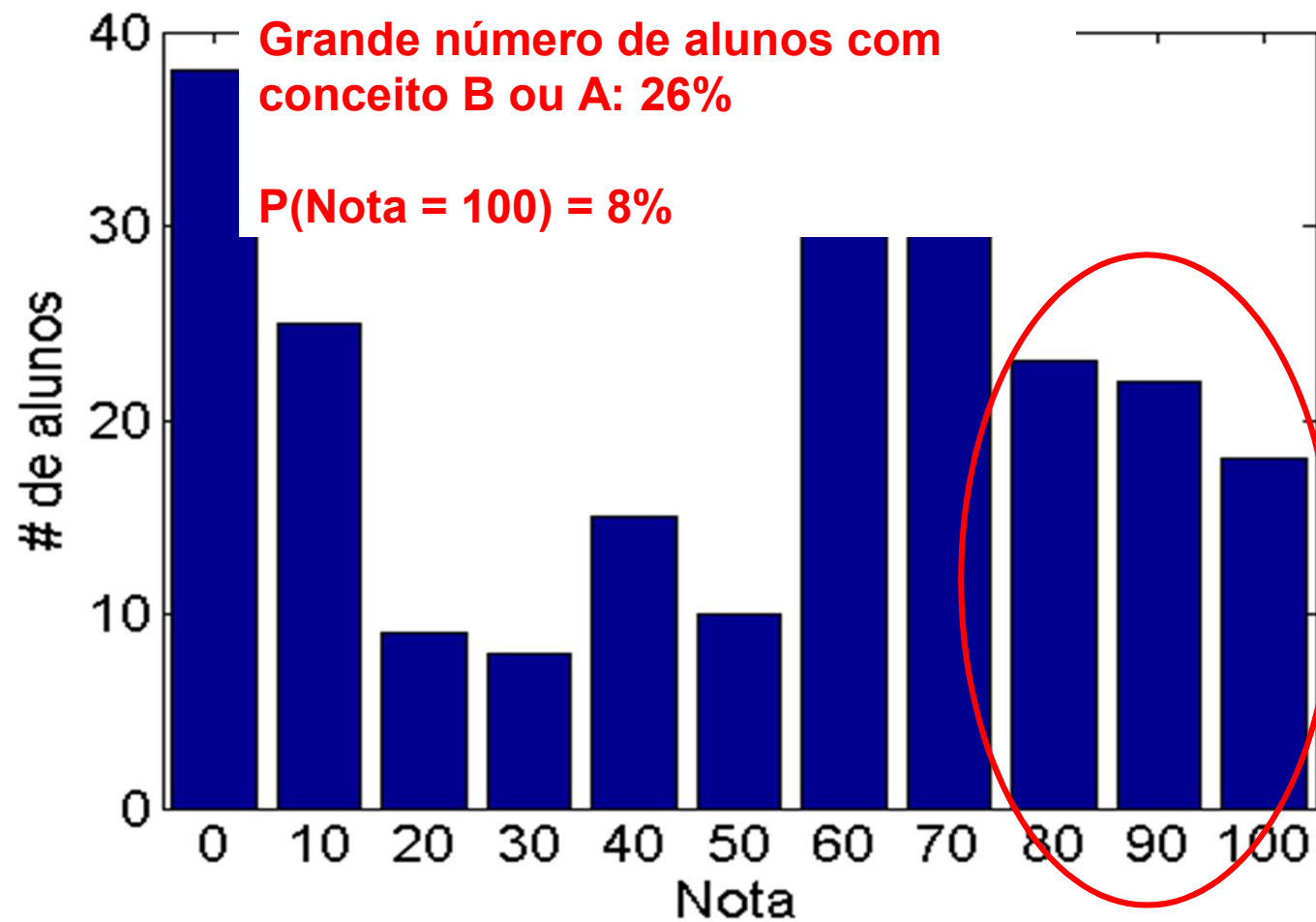
Desempenho dos Alunos



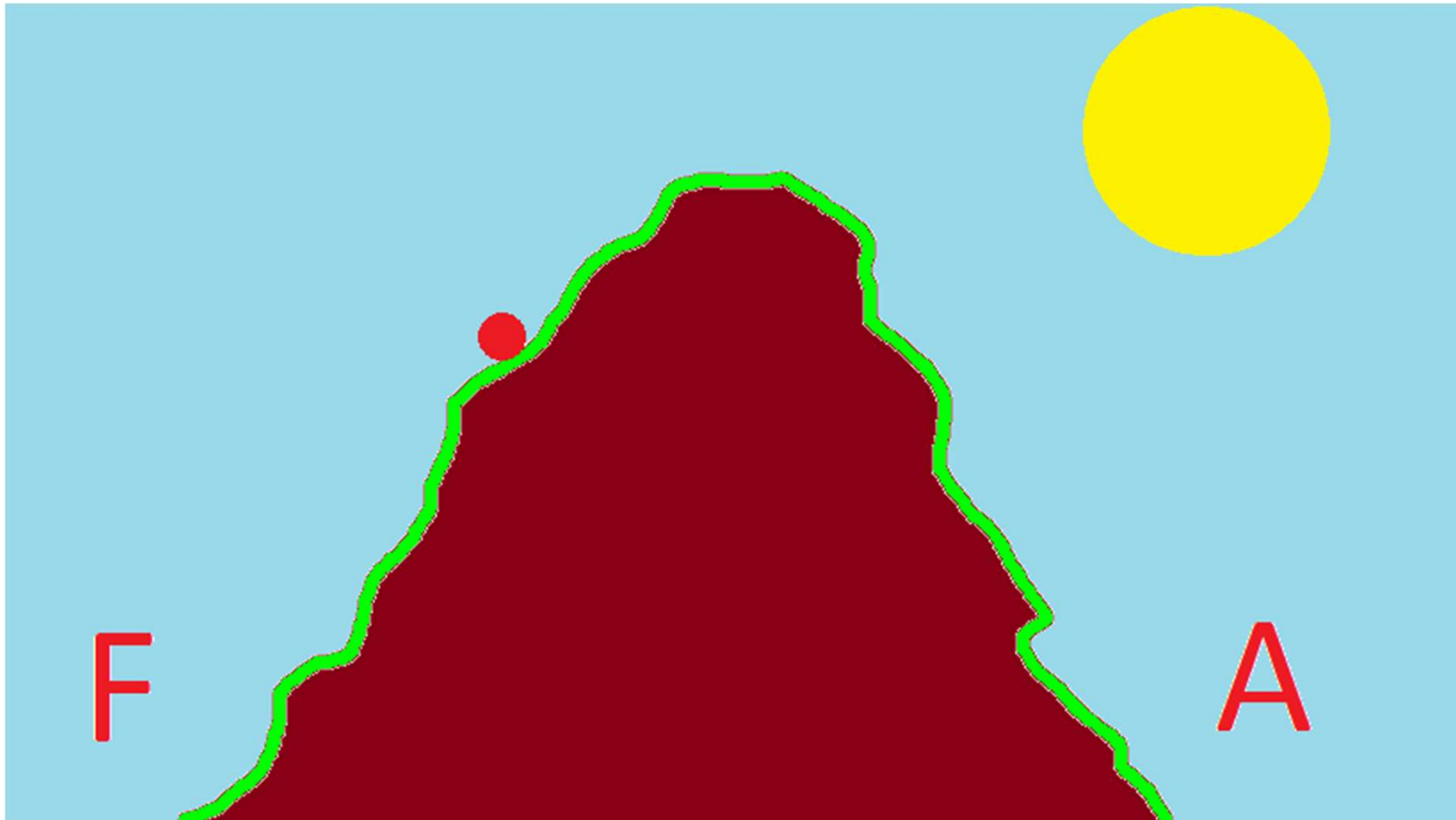
Desempenho dos Alunos



Desempenho dos Alunos



E você?



Observações

- Gostaria de aprovar todos, mas nem sempre isso é possível:
 - Programação é central em muitos cursos de exatas.
 - Vai ser importante para o seu futuro profissional.

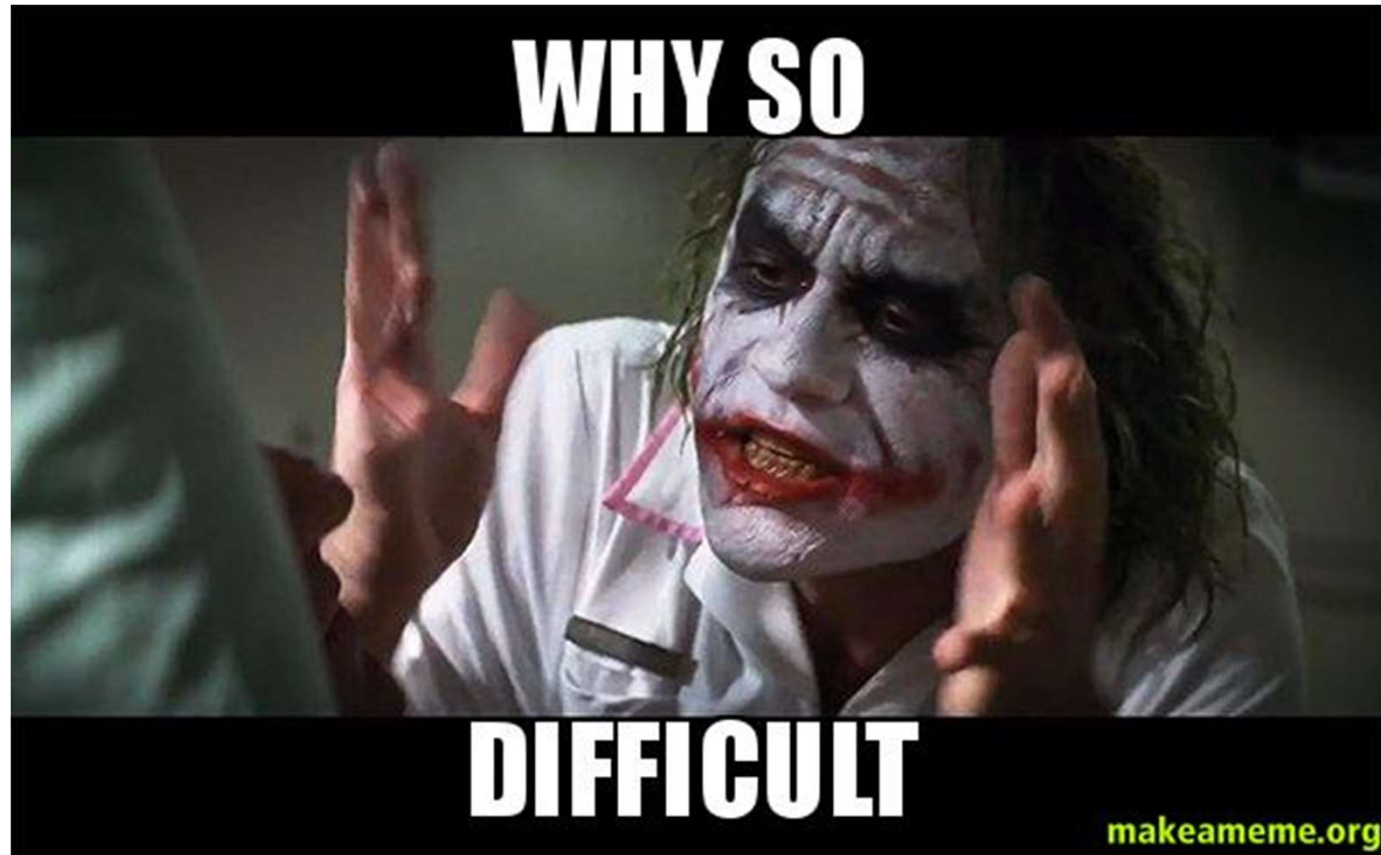
WHEN LIFE GIVES YOU FREE STUFF,



Práticas



Prova 1



Prova 2

Dicas



Prova 3

Dicas



TP

- Muitos desistem pois não conseguem entender NADA que está sendo dado em aula.
- CAUSA: Conteúdo é cumulativo, ou seja, para entender a aula x , é preciso ter entendido a aula $x-1$.
- Solução: prestar atenção
- Benefícios colaterais: menos estudo em casa, menos stress, menos fadiga.

Dicas



- Pergunte durante a aula!
 - Fico muito feliz com perguntas
 - Não existe pergunta boba
 - Eu **nunca** vou caçoar de uma pergunta feita por aluno
 - Então, não tenha medo de fazer perguntas bobas
 - A sua dúvida pode ser a dúvida de outro aluno

Dicas sobre o ERE

-
- É mais difícil verificar e acompanhar o aprendizado do aluno
 - Aluno terá maior responsabilidade no processo de aprendizado
 - Mais fácil trapacear e passar na disciplina
 - Mais fácil não aprender
 - Mais fácil atrasar a sua vida depois
 - Semestre passará muito rápido, cuidado!

Sobre corrupção em sala de aula: cola, compra e cópia de trabalhos

- Cola
 - Não é necessário pegar colando para identificar uma cola
 - Programas são como respostas dissertativas em português: estatisticamente, é impossível ter dois iguais!
 - Então, respostas estruturalmente iguais CARACTERIZAM uma cola

Sobre corrupção em sala de aula: cola, compra e cópia de trabalhos

- Compra e cópia de trabalhos
 - Será verificado na prova oral
 - Se você não conseguir responder sobre aspectos básicos do seu SUPOSTO programa, pontos serão retirados podendo chegar a 0.

Perguntas?

- E-mail: max.santiago@dcc.ufmg.br
 - . Por favor, evitem textos grandes.
- Material da disciplina: (**vai mudar**)
<https://github.com/MaxVieiraSantiago/UFMG/tree/master/PDS1>

Mensagem do Professor

Discussão aberta com os alunos

