Disciplina: Teoria da Computação N
Código: INF05501
Turma: A
Professor: Rafael Santos Coelho
Entrega: até 22/09/2019
Data de divulgação: 28/08/2019

INSTITUTO DE INFORMÁTICA UFRGS

TRABALHO DE PROGRAMAÇÃO EM MÁQUINA NORMA

Questão 1: (2,5 pontos) `[11.mn]` Escreva um programa NORMA que receba de entrada um número natural n e retorne a quantidade de ocorrências do padrão de *bits* 11 (**possivelmente com sobreposição**) na representação binária do número n. Seguem alguns casos de teste:

- seu programa ao receber de entrada o número 0 (que, em binário, é 0), deve retornar 0;
- seu programa ao receber de entrada o número 1 (que, em binário, é 1), deve retornar 0;
- seu programa ao receber de entrada o número 7 (que, em binário, é 111), deve retornar 2 (pois há 2 ocorrências do padrão de *bits* 11 em 111, a saber a ocorrência formada pelo primeiro *bit* e pelo segundo *bit* da esquerda para direita e a ocorrência formada pelo segundo *bit* e pelo terceiro *bit* da esquerda para direita);
- seu programa ao receber de entrada o número 62 (que, em binário, é 111110), deve retornar 4;
- seu programa ao receber de entrada o número 219 (que, em binário, é 11011011), deve retornar 3.

Questão 2: (2,5 pontos) `[mmc.mn]` Escreva um programa NORMA que compute a função $mmc : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ definida para todo par $(x, y) \in \mathbb{N} \times \mathbb{N}$ como descrito abaixo:

$$mmc(x,y) = \begin{cases} 0 & \text{se } x = 0 \text{ ou } y = 0, \\ \text{mínimo múltiplo comum de } x \text{ e } y & \text{caso contrário.} \end{cases}$$

Lembre-se de que, para quaisquer números naturais $x > 0$ e $y > 0$, o mínimo múltiplo comum de x e y é o menor número natural **positivo** divisível por x e por y. O par de entrada (x, y) vai estar sempre codificado de acordo com a codificação

$$(x, y) \mapsto 2^x(2y + 1) - 1.$$

Seguem alguns casos de teste:

- seu programa ao receber de entrada o número 1 (que codifica o par $(1, 0)$), deve retornar 0 (pois $mmc(1, 0) = 0$);
- seu programa ao receber de entrada o número 87 (que codifica o par $(3, 5)$), deve retornar 15 (pois $mmc(3, 5) = 15$);
- seu programa ao receber de entrada o número 575 (que codifica o par $(6, 4)$), deve retornar 12 (pois $mmc(6, 4) = 12$).

Questão 3: (2,5 pontos) `[pascoa.mn]` Escreva um programa NORMA que receba de entrada um número natural n (representando um ano d.C.) e retorne um número que codifica a data do feriado de Páscoa[1] no ano n no formato (dia, mês), onde $1 \leq$ dia ≤ 31 e $1 \leq$ mês ≤ 12. Para fins de teste, seu programa deve considerar **apenas anos do século XX** (ou seja, o número de entrada n sempre respeitará a condição $1901 \leq n \leq 2000$). Os passos do algoritmo para o cálculo da data estão descritos abaixo (comandos do tipo a mod b indicam o resto da divisão inteira de a por b e comandos do tipo a div b indicam a divisão inteira de a por b):

1. $a = n \bmod 19$
2. $b = n \operatorname{div} 100$
3. $c = n \bmod 100$
4. $d = b \operatorname{div} 4$
5. $e = b \bmod 4$
6. $f = (b + 8) \operatorname{div} 25$
7. $g = (b + 1 - f) \operatorname{div} 3$
8. $h = (19a + b + 15 - d - g) \bmod 30$
9. $i = c \operatorname{div} 4$
10. $k = c \bmod 4$
11. $\ell = (32 + 2e + 2i - h - k) \bmod 7$
12. $m = (a + 11h + 22\ell) \operatorname{div} 451$
13. mês $= (h + \ell + 114 - 7m) \operatorname{div} 31$
14. dia $= ((h + \ell + 114 - 7m) \bmod 31) + 1$

[1] A Páscoa, ao contrário do Natal, é um feriado "móvel", isto é, não possui uma data fixa. De acordo com a tradição, a Páscoa acontece no primeiro domingo após a primeira Lua cheia a partir do dia 21 de março.

O par de saída $(\text{dia}, \text{mês})$ deve ser codificado de acordo com a codificação

$$(\text{dia}, \text{mês}) \mapsto \frac{1}{2}(\text{dia} + \text{mês})(\text{dia} + \text{mês} + 1) + \text{mês}.$$

Seguem alguns casos de teste:

- seu programa ao receber de entrada o número 1901, deve retornar 70 (que codifica o par $(7, 4)$, isto é, 7 de abril);
- seu programa ao receber de entrada o número 1935, deve retornar 329 (que codifica o par $(21, 4)$, isto é, 21 de abril);
- seu programa ao receber de entrada o número 1997, deve retornar 564 (que codifica o par $(30, 3)$, isto é, 30 de março).

Para mais exemplos, acesse o link.

Dica: na resolução desta questão, pode ser bastante útil (e importante) usar os comandos `add`, `sub` e `cmp` do simulador. Para mais informações, acesse o link.

Questão 4: (2,5 pontos) `[rec.mn]` Escreva um programa NORMA que compute a função $rec : \mathbb{N} \to \mathbb{N}$ definida para todo $n \in \mathbb{N}$ da seguinte maneira:

$$rec(n) = \begin{cases} 0 & \text{se } n = 0, \\ 1 & \text{se } n = 1, \\ 3 & \text{se } n = 2, \\ 2 \cdot rec(n-1) + 3 \cdot rec(n-2) & \text{caso contrário.} \end{cases}$$

Seguem alguns casos de teste:

- seu programa ao receber de entrada o número 2, deve retornar 3 (pois $rec(2) = 3$);
- seu programa ao receber de entrada o número 4, deve retornar 27 (pois $rec(4) = 27$);
- seu programa ao receber de entrada o número 5, deve retornar 81 (pois $rec(5) = 81$).

Avisos importantes

- Este trabalho **deve** ser feito em grupos de **até** 5 pessoas. Trabalhos feitos por grupos que não se enquadrarem nessas condições ficarão automaticamente com nota 0. Trabalhos entregues com atraso (não importa quão pequeno seja o atraso ou o motivo do atraso) ficarão automaticamente com nota 0. **Nenhuma** tolerância com plágios: plagiados e plagiadores ficarão automaticamente com nota 0.
- O trabalho **deve** ser feito usando o simulador de máquina NORMA programado pelo professor Rodrigo Machado. Para cada questão do trabalho, o nome do código-fonte deve ser **exatamente** como está escrito no enunciado da questão (destacado

em vermelho). Por exemplo, o código-fonte relativo à Questão 1 **deve** se chamar `11.mn`, o código-fonte relativo à Questão 2 **deve** se chamar `mmc.mn` (repare que não há acentos, cedilhas ou letras maiúsculas; observe também que a extensão do arquivo é `.mn`) e assim por diante. Ao todo, são 4 arquivos do tipo `.mn`. Trabalhos que não respeitarem essa condição sofrerão uma penalidade de 0,1 ponto na nota para cada nome de código-fonte diferente do especificado.

- Códigos com erros sintáticos ficarão automaticamente com nota 0.
- O trabalho **deve** ser submetido na página da disciplina no Moodle. Essa é a **única** forma de submissão aceita. Trabalhos entregues via e-mail (ou via qualquer outro modo) não serão aceitos e ficarão automaticamente com nota 0. Na hora da submissão, compacte seus códigos-fontes e um arquivo `.txt` (tanto faz o nome do arquivo) contendo os nomes completos dos integrantes do grupo e seus respectivos números de cartão em um único arquivo do tipo `.zip` (tanto faz o nome do arquivo) e submeta esse arquivo compactado no Moodle. Trabalhos submetidos com arquivos compactados em qualquer extensão que não seja `.zip` terão um desconto na nota de 0,1 ponto. Se seu arquivo compactado estiver corrompido, o trabalho ficará automaticamente com nota 0. Apenas um membro de cada grupo deve fazer a submissão no Moodle em nome de todo o grupo. Se houver múltiplos envios por grupo (feitos por membros diferentes do mesmo grupo), o professor vai escolher um dos envios e só corrigirá o envio escolhido.
- A correção do trabalho será feita como descrito a seguir. Para cada questão, serão executados alguns testes (o número de testes por questão vai ser fixado posteriormente). O tempo limite de execução por caso de teste será de 100000 (100 mil) passos do simulador (com exceção da Questão 3, que terá tempo limite de execução de 1 milhão de passos do simulador). Para cada caso de teste com resposta errada ou que ultrapassar o tempo limite prescrito, será subtraído da nota $(2,5)/n$ ponto, onde n é o número total de testes feitos para a referida questão.
- Insira comentários nos seus códigos para deixá-los mais legíveis.
- Ao resolver o trabalho, prefiram algoritmos mais simples (e, nesse caso, simplicidade não necessariamente é sinônimo de elegância). Dependendo da questão, às vezes, um algoritmo força-bruta basta. Lembrem-se de que haverá um limite de tempo na execução dos testes. Por exemplo, se a solução que você pensou para alguma questão exige que você implemente codificação/decodificação de arranjos, muito provavelmente seu código vai ser demasiadamente lento e, sendo assim, corre o risco de estourar o limite de tempo.