Tarefa – Strings & malloc

Considere as informações dos candidatos inscritos em um concurso. Cada candidato recebe uma identidade secreta, que é o que deve constar na prova desse candidato.

A identidade secreta é uma cadeia de caracteres gerada com a seguinte lei de formação:
os 8 primeiros caracteres correspondem à data de nascimento (ano, mês, dia), os caracteres seguintes correspondem às iniciais do nome do candidato e após um *, vem o primeiro nome da mãe do candidato.

Exemplo: para o candidato chamado Ralph Gordon Lovelace, nascido em 24/10/1987, filho de Ada Lovelace, teríamos a seguinte cadeia: "19871024RGL*Ada".

Considere que os nomes têm apenas um branco entre cada uma das palavras que constituem um nome e não têm caracteres acentuados (tais como: ã, ç, ...). Também considere que os nomes não têm conectores do tipo "de", "dos", "da" e similares.

Faça um programa que solicita três informações do usuário (uma informação de cada vez e via teclado): o nome completo do candidato, a data de nascimento (no formato dd/mm/aaaa) e o nome completo da mãe. Depois escreva as seguintes funções:

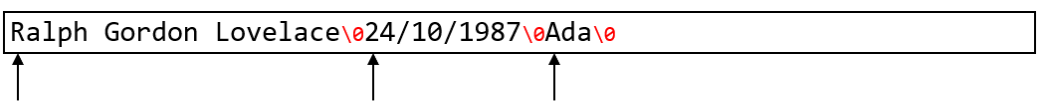
- uma função que recebe o nome completo do candidato, a data no formato dd/mm/aaaa e o primeiro nome da mãe e retorna uma nova *string* com a identidade secreta. Essa *string* deve ser alocada dinamicamente e do tamanho exato.
- uma função que, dada uma identidade secreta, retorna o primeiro nome da mãe (sem criar novo espaço de memória).
- uma função que, dada uma identidade secreta, imprime a data no formato dd/mm/aaaa.

Escreva, obrigatoriamente, as seguintes funções auxiliares:

- uma função que, dado um nome completo de candidato, retorna uma nova *string*, alocada dinamicamente e do tamanho exato, que contém as iniciais do nome.
- uma função que, dada uma data no formato dd/mm/aaaa, retorna uma nova *string*, alocada dinamicamente e do tamanho exato, que contém a data no formato aaaammdd.

Em todas as funções pedidas, caso não seja possível criar uma nova *string*, a função deve retornar NULL (apenas a *main* imprime mensagens de falta de memória). Trate do caso de insuficiência de memória e libere memória tão logo possa. Faça todos os testes que achar necessários.

Esta tarefa tem um desafio extra obrigatório: use apenas aritmética de ponteiros (não use a sintaxe de colchetes) e use apenas memória alocada dinamicamente (i.e. não é permitida alocação estática para nenhuma *string*). E, para ler dados do teclado, use apenas uma (e somente uma) grande área de memória alocada dinamicamente (onde você vai colocando as leituras do teclado). Por exemplo, após a leitura dos dados via teclado, essa área contém os seguintes caracteres e os seguintes ponteiros:



Após as leituras do teclado, essa área não deve ser alterada até o final do programa (quando, então. é liberada).

Faça *upload* do programa **.c** e de um arquivo **.txt** com a saída gerada pela execução do programa. Use, obrigatoriamente, dados completamente diferentes dos usados nos exemplos acima. Nomeie ambos os arquivos com o número da tarefa, sua turma, nome, matrícula e extensão correspondente, como no exemplo:

- T07_33X_MariaPatinhas_8752257.c
- T07_33X_MariaPatinhas_8752257.txt