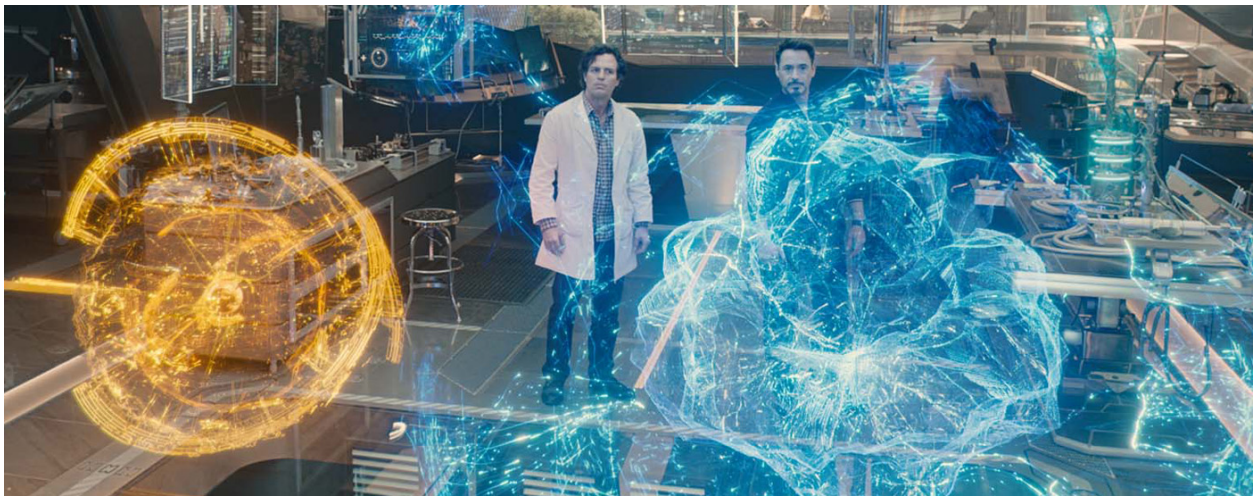
MC202AB - Estrutura de Dados

Lab 1 - Vetores e inteiros

Data da Primeira Chance: 29 de agosto de 2022
Peso: 1

Tony Stark está planejando construir um exército de robôs autônomos com um tipo de consciência única, o chamado Projeto Ultron. Para isso, ele vai precisar da ajuda de Bruce Banner, que também é um cientista extraordinário.

O plano de Tony é modelar todo o sistema autônomo com base em um artefato alienígena encontrado em uma missão, que parece funcionar na forma de neurônios, e nisso entra a ajuda de Dr. Banner, porque, com seu conhecimento, ele pode ajudar a colocar essa estrutura do artefato no sistema.



Porém, a *expertise* de Bruce é na área de biologia molecular e radiação gama (Hulk nasceu desse conhecimento), então ele não sabe muito sobre algoritmos e programação. Portanto, Tony se propôs a ajudar Bruce a entender melhor como trabalhar com estruturas de dados de forma mais eficiente sem deixar de fugir da sua área de pesquisa.

Tony propôs que Banner escreva um código em C que faz as seguintes tarefas:

- Ler um valor T $(4 \leq T \leq 1000)$, seguido dos valores inteiros $(n_1, n_2, \ldots n_T)$, correspondentes a T valores calculados por Tony e armazenar tais valores em um vetor u
- Implementar as seguintes operações para elas serem executadas conforme solicitado (não necessariamente nesta ordem):

1. **Conjunto de Banner**

    - Ler o valor inteiro ***B*** (*4 ≤ B ≤ 1000)* seguido dos valores ***(b_1, b_2, … b_B)*** correspondentes a ***B*** valores que fazem parte do chamado *Conjunto de Banner* (uma das criações científicas dele)

    - Modificar o vetor ***u*** de forma que os valores de ***u*** que não pertençam ao *Conjunto de Banner* sejam alterados para o valor zero

    - Imprimir na tela o vetor ***u*** modificado

2. **Multiplicação de Mobius**

    - Ler o valor ***M*** *(2 ≤ M ≤ 100)* que representa a constante para executar a *Multiplicação de Mobius*

    - Modificar o vetor ***u*** de forma que os valores nas posições ímpares serão multiplicadas por ***M***

    - Imprimir na tela o vetor ***u*** modificado

3. **Movimento Cíclico**[1]

    - Ler o valor ***P*** *(1 ≤ P < T)* que representa o tamanho do passo a ser feito no *Movimento Cíclico* que Bruce precisa fazer no vetor ***u***

    - Modificar o vetor ***u*** de forma que os valores serão um *movimento cíclico* dele mesmo

    - Imprimir na tela o vetor ***u*** modificado

      Exemplos de movimento cíclico:

      ```
      Ex.1: O vetor (1, 2, 3, 4, 5, 6, 7) se torna o vetor (5, 6, 7, 1, 2, 3, 4) com um passo de tamanho 3, isto é, o valor na última posição do vetor foi movido para a primeira posição 3 vezes.

      Ex.2: O vetor (1, 2, 3, 4, 5, 6, 7) se torna o vetor (7, 1, 2, 3, 4, 5, 6) com um passo de tamanho 1

      Ex.3: O vetor (0, 0, 0, 1, 0, 1, 0, 0) se torna o vetor (0, 0, 0, 0, 1, 0, 1, 0) com um passo de tamanho 1
      ```

[1] Movimento cíclico neste laboratório é a operação de reorganizar os valores de um vetor, movendo o valor da última posição para a primeira posição, deslocando todos os outros valores para a posição seguinte a dele.

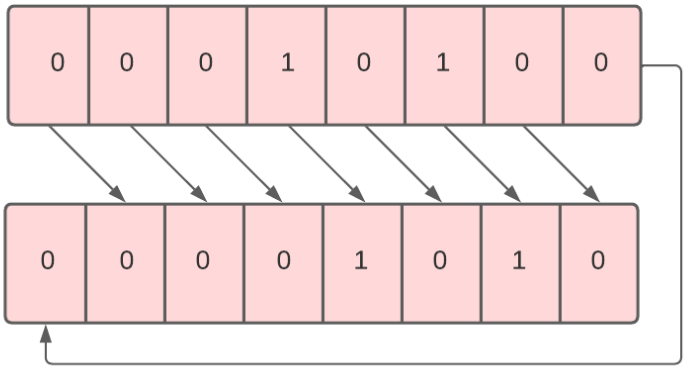


Figura 1: Exemplo 3 - movimento cíclico

4. **Inversão de vetor**
    - Modificar o vetor ***u*** de forma que os valores estejam em ordem inversa, para então Tony Stark poder continuar a sua parte de sequenciamento da consciência central do sistema Ultron. Exemplo: vetor (1, 2, 3, 4) se torna (4, 3, 2, 1).
    - Imprimir o vetor ***u*** modificado

Dr. Banner precisa fazer essas tarefas com muito cuidado e atenção com o desempenho de seu código, pois são mais de 2 trilhões de neurônios artificiais a serem programados por ele e Tony Stark por meio desses vetores da tarefa, e um cálculo errado pode causar uma cascada de falhas.

Sua tarefa, como aluno de MC202, é fazer o papel do Dr. Banner e realizar as modificações no vetor ***u*** conforme as operações solicitadas na entrada do programa.

Observação: Os intervalos dos valores de T, B, P e M não precisam ser verificados no código, isto é, não é necessário fazer "`if (T >= 4 && T <= 1000)`", por exemplo. Todos os valores estarão dentro do intervalo informado.

Entrada

A entrada será fornecida da seguinte forma:

- Valor ***T***, representando a quantidade de valores a serem disponibilizados em seguida
- Na linha seguinte, uma quantidade ***T*** de valores inteiros fornecidos por Tony Stark, disponibilizados em uma única linha e separados por ' ' (espaço em branco)
- Na linha seguinte, a quantidade de operações que serão feitas

- Nas linhas seguintes (cada operação pode ser executada várias vezes e em várias ordens), o número correspondente da operação e seus respectivos parâmetros. Note que:
    - Operação 1: Número da operação + ***B*** + valores do *Conjunto de Banner*. Ex.:

      ```
      1 11 12 15 78 52 48 68 57 9 2 35 77
      ```

    - Operação 2: Número da operação + ***M***. Ex.:

      ```
      2 45
      ```

    - Operação 3: Número da operação + ***P***. Ex.:

      ```
      3 4
      ```

    - Operação 4: Número da operação somente. Ex.:

      ```
      4
      ```

Saída

Todos os vetores de saída devem ser impressos em uma única linha, separando valores por espaço. Dito isto, a cada operação realizada, o vetor resultante dessa operação deve ser impresso.

Dica: Use **funções**.
Atenção: O vetor ***u*** modificado em uma operação é o mesmo vetor que será utilizado na operação seguinte, então ao final do programa, o mesmo vetor ***u*** terá sido modificado várias vezes de forma sequencial

Exemplos

Exemplo 1:

Entrada

```
5
1 2 3 4 5
3
1 4 4 9 2 1
3 2
4
```

Saída

```
1 2 0 4 0
4 0 1 2 0
0 2 1 0 4
```

Exemplo 2:

Entrada

```
8
5 31 12 9 4 52 1 99
4
2 10
2 2
1 5 180 201 605 12 620
3 1
```

Saída

```
5 310 12 90 4 520 1 990
5 620 12 180 4 1040 1 1980
0 620 12 180 0 0 0 0
0 0 620 12 180 0 0 0
```

Regras e Avaliação

Seu código será avaliado não apenas pelos testes do CodePost, mas também pela qualidade. Dentre os critérios subjetivos de qualidade de código iremos analisar: o uso apropriado de funções, de comentários, e de structs; a escolha de bons nomes de funções, variáveis e de structs e seus campos; o correto uso de Tipos Abstratos de Dados e a correta separação em vários arquivos; a ausência de diversos trechos de código repetidos, e o tempo de execução e uso de memória dos algoritmos projetados. Note, porém, que essa não é uma lista exaustiva, pois outros critérios podem ser analisados dependo do código apresentado visando mostrar ao aluno como o código poderia ser melhor.

Submissão

Você deverá submeter no CodePost, na tarefa Lab 01 - *Vetores e inteiros*, um arquivo com o nome `lab01.c`. Após o prazo estabelecido para a atividade, será aberta uma tarefa Lab 01 - Segunda Chance, com prazo de entrega até o fim do semestre.