Monitoria PROG II

Lista 2 - Exercício 1

Exercício:

Vamos trabalhar o conceito de ponteiro de ponteiros. Para isso, vamos criar um tipo Vector que contenha um outro tipo VectorItem, que por sua vez abriga um ponteiro para int (int *). Ambas as estruturas deverão ser opacas.

A struct VectorItem deverá possuir a seguinte forma:

```
struct VectorItem{
    int *data;
};
```

Além disso, a struct VectorItem deverá possuir as seguintes funções:

- **VectorItem *vector_item_create(int *data)** - Essa função aloca espaço para o VectorItem e armazena um ponteiro de int;
- **void vector_item_print(VectorItem *item)** – Essa função imprime o valor de int do VectorItem na tela;
- **void vector_item_free(VectorItem *item)** - Essa função desaloca o espaço alocado para o VectorItem;

Em seguida, deve-se criar o tipo Vector, que abrigará um array de elementos do tipo VectorItem. Note que, por ambas as estruturas serem opacas, o tipo Vector conterá um ponteiro de ponteiros para VectorItem.

Lembre-se também de armazenar o tamanho atual e o tamanho alocado do seu vetor no tipo Vector. Essas informações serão úteis na implementação das funções.

Definidos os tipos listados acima, você deverá implementar as seguintes funções:

- **Vector *vector_create() -** Aloca espaço para o tipo Vector e retorna um ponteiro para esse tipo.
- **void vector_add(Vector v, int *elem) -** Adiciona um ponteiro do tipo int ao vetor. Note que aqui talvez será preciso realocar o espaço, caso o número de elementos dentro do vetor seja maior do que o número de elementos alocados.
- **VectorItem vector_get(Vector *v, int idx) -** Retorna o item do tipo VectorItem de um determinado índice do vetor.
- **int vector_size(Vector *v)** - Retorna o tamanho do vetor.
- **void vector_remove(Vector *v, int idx)** – Remove o elemento de um determinado índice. Note que, ao remover o elemento e destruí-lo, a posição ficará vazia no vetor. Portanto, você deve mover o elemento de idx + 1 para idx, de idx + 2 para idx + 1 e assim por diante.
- **Void vector_print(Vector *v)** – Imprime os elementos de um vetor na tela.
- **void vector_destroy(Vector *v)** - Desaloca o espaço alocado para o tipo Vector.

A main que deverá ser usada é a seguinte:

```c
#include <stdio.h>
#include "vector.h"

int main(){
    Vector *v = vector_create();
    int qntdNumeros;

    scanf("%d", &qntdNumeros);

    for(int i = 0; i < qntdNumeros; i++){
        vector_add(v, &i);
        vector_print(v);
    }

    vector_remove(v, qntdNumeros - 1);
    vector_print(v);

    vector_destroy(v);
}
```

Entrada:	**Saída**
5	[0] [1, 1] [2, 2, 2] [3, 3, 3, 3] [4, 4, 4, 4, 4] [5, 5, 5, 5]