INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA BAHIA	**Curso: Análise e Desenvolvimento de Sistemas**
	Turno: Noturno / **Data: 28/08/2015**
	Disciplina: INF029 – Laboratório de Programação
Professor: Renato Novais	
Estudante:	**Nota:**

Avaliação I

1. Considere o código fonte disponível a seguir. Ele é uma implementação da atividade apresentada em sala, que cria um vetor de 10 posições, onde cada posição do vetor (endereçadas de 1 a 10) apontam para outro vetor, de tamanho definido pelo usuário. A Figura 1 apresenta um exemplo da estrutura resultante após o usuário ter inserido alguns valores.

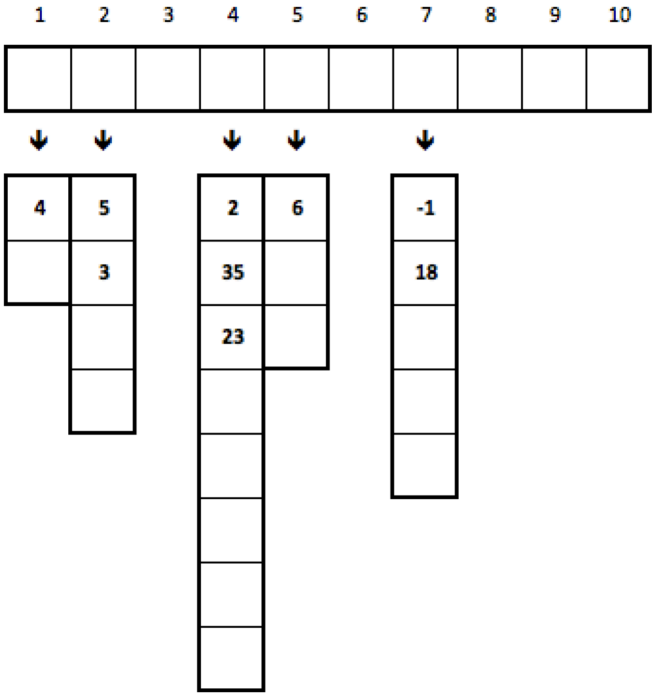

Figura 1 – Exemplo de entrada de dados

Não crie nenhum parâmetro global. Faça:

a. (Valor 1.5) Reimplemente a função **int verifica(int espaco)**, para que ela tenha tipo de retorno **void**, mas que continue atingindo o mesmo objetivo. Além da própria função destaque no código o que precisa ser mudado em função da mudança realizada em "**verifica**";

b. (Valor 2.5) A função **void liberarEspaco(Celula vetor[])** não está funcionando corretamente. Corrija-a, considerando que ela deveria liberar o vetor da posição informada pelo usuário;

c. (Valor 6.0) Implemente a opção 4 do menu principal. A ideia é imprimir os elementos do vetor da seguinte forma: primeiro imprima todos os primeiros elementos, depois todos os segundos, depois todos os terceiros, e assim sucessivamente. Para o exemplo da Figura 1, a saída deveria ser 4 5 2 6 −1 3 35 18 23. Caso precisa criar alguma estrutura auxiliar, lembre-se de liberar seu espaço no final do seu uso.

```c
#include<stdio.h>
#include<stdlib.h>
#include <locale.h>
#define dimencao 10

typedef struct celula{
    int *endereco;
    int cont;
    int tam;
}Celula;

void menu();
int  verifica(int espaco);
void liberarEspaco(Celula vetor[]);
void inicializa(Celula vetor[]);
void cria_vetor(Celula vetor[], int posicao);
void insere(Celula vetor[]);
void imprime(Celula vetor[]);

int main(){
    setlocale(LC_ALL, "Portuguese");
    Celula vetor[dimencao];
    int opcao;
    inicializa(vetor); // Inicializa todo o vetor com valor NULL
    menu();
    scanf("\n%d", &opcao);
    while(opcao != 0){
        switch(opcao){
            case 1: insere(vetor); break;
            case 2: liberarEspaco(vetor); break;
            case 3: imprime(vetor); break;

            default: printf("\nValor inválido!\n"); break;
        }
       menu();
       scanf("\n%d", &opcao);
    }
    return 0;
}

void menu(){
    printf("********************* MENU ************************");
    printf("\n*\tDigite uma das opções abaixo:              *");
    printf("\n*\tDigite 1 para inserir elementos no vetor.  *");
    printf("\n*\tDigite 2 para liberação de espaço no vetor.*");
    printf("\n*\tDigite 3 para imprimir todo o vetor.       *");
    printf("\n*\tDigite 4 Imprimir intercalado inicial      *");
    printf("\n*\tDigite 0 para SAIR.                        *");
    printf("\n---------------------------------------------------\n");
    printf("Digite AQUI -> ");
}

void liberarEspaco(Celula vetor[]){
    int espaco;
    printf("Informe o vetor que deseja liberar: Faixa entre 1 - 10:");
    printf(" -> ");
    scanf("\n%d", &espaco);
    espaco = verifica(espaco);
    if (vetor[espaco].endereco == NULL){
        printf("\nVetor %d VAZIO não é possível liberar espaço !!!\n\n", espaco);
    }else{
        vetor[espaco].endereco = NULL;
        vetor[espaco].cont =0;
        vetor[espaco].tam  =0;
        printf("\nVetor %d LIBERADO !!!\n", espaco);
    }
}

void inicializa(Celula vetor[]){

    int i;
    for (i=0; i<dimencao; i++){
        vetor[i].endereco = NULL;
    }
}

```

```c
void cria_vetor(Celula vetor[], int posicao){

    int size;
    printf("\nInforme o tamanho do vetor na posição %d: ", posicao);
    scanf("\n%d",&size);
    while(size <= 0){
        printf("\nTamanho deve ser maior que zero !!! ");  // validar a faixa
        printf("\nInforme novamente -> ");
        scanf("\n%d",&size);
    }
    vetor[posicao-1].endereco = (int *) malloc(size*sizeof(int));
    vetor[posicao-1].cont = 0;
    vetor[posicao-1].tam = size;
}

void insere(Celula vetor[]){

    int posicao, num;
    printf("\nQual posição você deseja inserir? Faixa entre 1 - 10: ");  // validar a faixa
    printf("-> ");
    scanf("\n%d",&posicao);
    posicao = verifica(posicao);
    if(vetor[posicao-1].endereco == NULL){
        cria_vetor(vetor, posicao);
    }
    if(vetor[posicao-1].cont == vetor[posicao-1].tam)    //verificar se o vetor endereço está cheio
        printf("\nVetor %d está CHEIO !!! \n\n", posicao);
    else{
        printf("\nInforme o elemento: ");
        printf("-> ");
        scanf("\n%d",&num);
        vetor[posicao-1].endereco[vetor[posicao-1].cont] = num;
        vetor[posicao-1].cont++;
        printf("\n");
    }
}

void imprime(Celula vetor[]){

    int i, j;
    for(i=0; i<dimencao; i++){
        if (vetor[i].endereco != NULL){
            printf("\n------------");
            printf("\nVetor %d", i+1);
            printf("\n____________");
            for(j=0; j<vetor[i].cont; j++)
                printf("\n%d", vetor[i].endereco[j]);
            printf("\n\n");
        }else
          printf("\nVetor %d VAZIO", i+1);
    }
    printf("\n\n");
}

int verifica(int espaco){

    while(espaco < 1 || espaco > 10){
        printf("\nPosição do vetor não corresponde! Faixa entre 1 - 10: ");
        printf("\nInforme novamente -> ");
        scanf("\n%d",&espaco);
    }
    return espaco;
}
```