GRADUAÇÃO EM ANÁLISE E DESENVOLVIMENTO
PROGRAMAÇÃO DE COMPUTADORES II
Prática 11: Ponteiros e Alocação Dinâmica

- *Cada programa deve ser efetuado em um arquivo "C" próprio contendo como nome ex1.c para o exercício 1, ex2.c para o exercício 2 e assim por diante.*

1. **Faça um programa em C que obtenha um valor do tipo *int* e outro do tipo *char*, armazene o endereço de memória destes valores em duas variáveis do tipo ponteiro e imprime a posição de memória apontada por cada ponteiro, bem como o conteúdo armazenado nessas posições de memória, utilizando para tal as duas variáveis ponteiro.**

2. **Faça um programa em C que obtenha dois valores inteiros, após isso armazene o endereço de memória dos dois valores em dois ponteiros e depois efetue a soma destes valores, utilizando para isso as duas variáveis ponteiro e armazenando o resultado em uma terceira variável inteira.**

3. **Faça um programa em C que obtenha o tamanho (inteiro) de uma frase (*string*). Após isso aloque dinamicamente (*malloc*) um ponteiro de caracteres com tamanho obtido e solicite ao usuário que digite uma frase a ser armazenada nesse vetor alocado.**

4. **Faça um programa em C que obtenha o tamanho de duas palavras (*string*). Após isso aloque (*malloc*) dois ponteiros com o tamanho especificado e solicite ao usuário que digite duas palavras, armazenando-as nos ponteiros alocados. Utilizando um terceiro ponteiro aloque-o com tamanho igual a soma do tamanho dos ponteiros anteriores mais um e armazene nele as duas frases anteriores concatenadas por um hífen. Exemplo:**

Tamanho da primeira palavra:	9
Primeira Palavra:	Instituto
Tamanho da segunda palavra:	4
Segunda Palavra:	IFPR
Concatenadas:	**Instituto-IFPR**

5. **Faça um programa em C que obtenha o tamanho de frase (*string*). Após isso aloque (*malloc*) um ponteiro com o tamanho especificado e solicite ao usuário que digite a frase armazenando-a no ponteiro alocado, para tal utilize a função *gets()*. Após guardar a frase no ponteiro codifique uma rotina que apresente o total de caracteres da frase digita.**

6. **Faça um programa em C que obtenha o tamanho de duas palavras (*string*). Após isso aloque (*malloc*) dois ponteiros com o tamanho especificado e solicite ao usuário que digite duas palavras, armazenando-**

as nos ponteiros alocados. Com as palavras armazenadas codifique uma rotina que calcule o total de caracteres das duas palavras. Agora aloque um terceiro ponteiro para armazenar as duas palavras concatenadas por um sinal de '+'.

7. **Faça um programa em C que obtenha o tamanho de uma palavra (*string*). Após isso aloque (*malloc*) dois ponteiros com o tamanho especificado, o primeiro para receber a palavra que será digitada pelo usuário e o segundo para onde irá copiar esta palavra só que de maneira invertida. Exemplo:**

Tamanho da palavra:	20
Palavra:	Gil Eduardo
Palavra Invertida:	**odraudE liG**

8. **Faça um programa em C que obtenha o tamanho de uma frase (*string*). Após isso aloque (*malloc*) dois ponteiros, um com o tamanho especificado e outro com metade deste tamanho. Solicite ao usuário que digite uma frase e armazene-a no primeiro ponteiro. Por fim copie para o segundo ponteiro apenas os caracteres que estão nas posições ímpares deste primeiro ponteiro e exiba-o. Exemplo:**

Tamanho da frase:	20
Frase:	Instituto Federal
Caracteres Ímpares:	**nttt eea**

9. **Faça um programa em C que obtenha o tamanho de duas palavras (*string*). Após isso aloque (*malloc*) dois ponteiros com os tamanhos especificados. Solicite ao usuário que digite duas palavras e armazene-as nos dois ponteiros. Agora efetue uma comparação entre as duas palavras, caractere a caractere, indicando se elas são exatamente iguais ou se são diferentes. Caso sejam diferentes especifique em qual posição e em quais letras se diferenciam. Exemplo:**

Tamanho da primeira palavra:	21
Primeira palavra:	improvável
Tamanho da segunda palavra:	40
Segunda palavra:	impossível
Diferentes:	**posição 3 - (r) diferente de (o)**

10. **(*DESAFIO*) Faça um programa em C que obtenha uma frase e armazene-a numa variável ponteiro alocada inicialmente com tamanho um. Neste exercício você não deve utilizar a função *gets()*, mas sim a implementar uma rotina similar a ela que permitirá ao usuário armazenar a frase digitada no ponteiro alocado com tamanho um. Para tal utilize a função *getchar()* que irá capturar caractere a caractere a frase digitada e realocar (*realloc*) o tamanho deste ponteiro um a um. A rotina deve terminar a capturar dos caracteres quando a tecla ENTER for pressionada. Ao final da captura exiba o tamanho e o conteúdo final do ponteiro.**