EA871 – Laboratório de Programação Básica de Sistemas Digitais

Atividade 02 – 2º semestre de 2023

1. Objetivos

- Familiarização com a linguagem C e algoritmos essenciais
- Utilizar ponteiros, vetores e funções

2. Resumo da Atividade

Nesta atividade, aprenderemos a desenvolver e analisar programas em C usando ponteiros, vetores e funções.

3. Roteiro da Aula

Exercício 1

Leia o código-fonte abaixo, que foi escrito em linguagem C.
a) Encontre todas as palavras-chave que você não conhece.
b) Faça uma hipótese sobre o que o programa faz.
c) Compile e execute o programa. Ele faz o que você imaginava?
d) O que significa void()?
e) Como a função procedimentoN() "sabe" que deve imprimir o valor 3 na tela?
f) Modifique o programa para que a função procedimentoN() imprima o valor 95, ao invés de 3.

```
#include <stdio.h>

void procedimento() {
	printf("Ola, mundo! Eu sou um procedimento!\n");
}

void procedimentoN(int N) {
	printf("Ola, mundo! Eu sou o procedimento %d!\n", N);
}

int main() {
	int i, j;
	printf("Procedimentos:\n");
	procedimento();
	procedimentoN(2);
	i = 3;
	procedimentoN(i);
	return 0;
}
```

Exercício 2

Leia o código-fonte abaixo e:
a) Encontre todas as palavras-chave que você não conhece.
b) Faça uma hipótese sobre o que o programa faz.
c) Compile e execute o programa. Ele faz o que você imaginava?
d) O rótulo *i* definido dentro da função procedimentoEscopo() se refere à mesma variável *i* definida na função main()? Como é possível demonstrar isso?

```
#include <stdio.h>

void procedimentoN(int N) {
	printf("Ola, mundo! Eu sou o procedimento %d!\n", N);
}
```

```
void procedimentoEscopo(int N) {
	int i;
	i = N + 1;
	printf("Sucessor de %d: %d\n", N, i);
}

int main() {
	int i;
	i = 3;
	procedimentoN(i);
	printf("----\n");
	printf("Escopo:\n");
	printf("i=%d (main)\n", i);
	procedimentoEscopo(i);
	printf("i=%d (main)\n", i);
}
```

Exercício 3

Ao declarar uma variável, estamos pedindo ao compilador que reserve um espaço de memória correspondente ao tipo (int, char, float) e então associe esse espaço ao rótulo que escolhemos. É possível descobrir o endereço de memória que foi alocado para nossa variável usando o operador **&**:

```
#include <stdio.h>

int main() {
	int i;
	printf("A variavel i tem valor %d e esta na posicao 0x%x\n", i, &i);
	return 0;
}
```

Também, podemos criar uma variável do tipo "ponteiro", que contém um endereço de memória. Para isso, usamos o modificador *, na forma:

```
#include <stdio.h>

int main() {
	int i;
	int *j; /* j eh um ponteiro para um int */
	i = 10;
	j = &i; /* j recebe o endereco de i */
	*j = 50; /* o endereco apontado por j recebe 50 */
	printf("i=%d\n", i);
	return 0;
}
```

O programa abaixo contém uma função que recebe ponteiros como parâmetros.
a) Encontre todas as palavras-chave que você não conhece.
b) Faça uma hipótese sobre o que o programa faz.
c) Compile e execute o programa. Ele faz o que você imaginava?

```
#include <stdio.h>

void procedimento_ref(int *N, int M) {
	*N = *N + 1;
	M = M + 1;
}

int main() {
	int i, j;
	i = 0;
```

```
    j = 0;
    procedimento_ref(&i, j);
    printf("i=%d, j=%d\n", i, j);
    return 0;
}
```

Exercício 4

Colchetes [] são usados em C para definir e acessar vetores. Ao declarar um vetor, como em:

```
int i [10];
```

estamos pedindo ao compilador que reserve espaço suficiente para um certo número (neste caso, 10) de variáveis do tipo definido (neste caso, int).

Podemos então acessar cada posição usando:

```
printf("%d\n", i[0]);
printf("%d\n", i[3]);
printf("%d\n", i[4]);
```

Lembre-se que a posição inicial tem índice 0 e que, portanto, a posição final do vetor tem índice 9 (neste caso, já que definimos um vetor de 10 posições).

Analise o programa abaixo e:
a) Encontre todas as palavras-chave que você não conhece.
b) Faça uma hipótese sobre o que o programa faz.
c) Compile e execute o programa. Ele faz o que você imaginava?

```
#include <stdio.h>

#define TAMANHO_VETOR 5

void imprimir_vetor(int vetor[], int N) {
    int i;
    for (i=0; i<N; i++) {
        printf("%d\t", vetor[i]);
    }
    printf("\n");
}


void ler_vetor(int vetor[], int N) {
    int i;
    int j;
    for (i=0; i<N; i++) {
        scanf("%d", &j);
        vetor[i] = j;
    }
}


int produto_interno(int vetor1[], int vetor2[], int N) {
    int p;
    int i;
    p = 0;
    for (i=0; i<N; i++)
        p = p + (vetor1[i] * vetor2[i]);

    return p;
```

```
}

int main() {
    int vetor_base[] = {1, 2, 6, 3, 5};
    int meu_vetor[TAMANHO_VETOR];
    ler_vetor(meu_vetor, TAMANHO_VETOR);
    printf("Vetor lido! Imprimindo...\n");
    imprimir_vetor(meu_vetor, TAMANHO_VETOR);
    printf("Produto interno com vetor-base: %d\n", produto_interno(vetor_base, meu_vetor, TAMANHO_VETOR));
    return 0;
}
```

Exercício 5

No programa abaixo,
a) Faça uma hipótese sobre o que o programa faz.
b) Compile e execute o programa. Ele fez o que você imaginava?

```
#include <stdio.h>


int main() {
    char string[] = {'a', 'b', 'c', 'd', 'e'};
    char *p1 = &(string[0]);
    char *p2 = string;
    for (int i=0; i<5; i++) {
        printf("*p1=%c\t", *p1);
        p1++;
        printf("*p2=%c\n", *p2);
        p2++;
    }
    return 0;
}
```

4. Exercício computacional para casa (individual)

O objetivo desse exercício é manipular um vetor que armazena uma cadeia de caracteres, uma *string*. O texto é um trecho do discurso de promulgação da Constituição de 1988:

A Constituição certamente não é perfeita. Ela própria o confessa ao admitir a reforma. Quanto a ela, discordar, sim. Divergir, sim. Descumprir, jamais. Afrontá-la, nunca.
Traidor da Constituição é traidor da Pátria. Conhecemos o caminho maldito. Rasgar a Constituição, trancar as portas do Parlamento, garrotear a liberdade, mandar os patriotas para a cadeia, o exílio e o cemitério.
Quando após tantos anos de lutas e sacrifícios promulgamos o Estatuto do Homem da Liberdade e da Democracia bradamos por imposição de sua honra.
Temos ódio à ditadura. Ódio e nojo.
Amaldiçoamos a tirania aonde quer que ela desgrace homens e nações. Principalmente na América Latina.

Ulysses Guimarães, Presidente da Assembleia Nacional Constituinte.

Com base no arquivo "Atividade_02.txt", os alunos devem completar o programa de modo a implementar a função *conta_palavras* que recebe o texto e retorna o número de palavras nele (palavras separadas por hífen contam como duas). Devem completar também a função *imprime_palavras_c* que imprime todas as palavras que se iniciam com a letra *c* na sequência em que aparecem. Elas devem ser impressas uma por linha. Não utilizem funções prontas para manipular cadeias de caracteres. **A única biblioteca que pode ser usada é a *stdio.h*.**
DICA: O último caractere da cadeia e que indica o seu final é o caractere nulo, '\0'.

Instruções para a submissão do trabalho

1) Baixe o *template* da atividade 2 do Google Classroom.
2) Modifique o arquivo Atividade_02.txt para completar seu laboratório.
3) Comente o programa descrevendo a lógica que foi elaborada para contar e imprimir as palavras. Caprichem nos comentários, eles constituem o relatório de vocês.
4) Quando terminar, crie um arquivo cujo nome é seu_ra.txt (Exemplo: 025304.txt).
5) Faça o *upload* da sua solução da atividade 2 no Google Classroom.