Estrutura e Recuperação de Dados A

Atividade 1

Ponteiros

Grupo:

Bruno Camilo.....RA:16080293

Iago Lourenço.....….....RA:15610116

Exercício 1

```
#include<stdio.h>

#include<stdlib.h>


void troca ( int *a , int * b )

{

	int temp ;

	temp = *a ;

	*a = *b ;

	*b = temp;

}

int main ()

{

	int x , y ;

	scanf ( "%i %i",&x ,&y );

	troca (&x , &y );

	printf ( "Troquei ----> %i %i\n" , x , y );

	return 0;

}
```

Exercício 2

As variáveis *i* e *j* são declaradas respectivamente 10 e 20, a pós ter seus valores declarados as duas variáveis tem seu endereço de memoria passado para *p1* e *p2* respectivamente. Com os endereços das variáveis, ocorre uma troca entre os valores contido em *p1* e *p2* com ajuda de uma variável *temp*, portanto oque será impresso são os valores trocados uma vez que o endereço de memoria que a variável *i* aponta foi passado para a variável *j*, o endereço de memoria que a variável *j* aponta foi passado para a variável *i.* Dessa forma *i* agora contem 20 e *j* contem 10

```
#include <stdio.h>

#include <stdlib.h>

int main ()

{

        int i = 10 , j = 20;

        int temp;

        int *p1 , *p2;

        p1 = &i ;

        p2 = &j;

        temp = *p1 ;

        *p1 = *p2 ;

        p2 = temp ;

        printf ( " %d %d \n " , i , j );

return 0;

}
```

Exercício 3

```
#include<stdlib.h>

#include<stdio.h>

void Troca ( int *a , int *b )

{

        int temp ;

        temp = *a ; *a = *b ; *b = temp ;

}

int main ()

{

        int x , y ;

        int *px , *py ;

        px = &x ;

        py = &y ;


        scanf ( "%d %d" , px, py);

        Troca ( px, py);

        printf ( "Troquei ----> %d %d \n",x,y );

        return 0;

}
```

Exercício 4

O programa ira imprimir o Sobrenome declarado pelo usuário. Isso acontece, pois o nome completo é passado para a função e lá ela encontra onde começa o sobrenome e guarda a posição de memoria desse sobrenome na variável *pnome*. Após ter encontrado o sobrenome a função retorna a posição para a variável *p* que então printa o sobrenome na tela.

```
#include<stdio.h>

#include<stdlib.h>

char * acheSobrenome(char nome[])

{

	char *pnome;

	int	i = 0;


	while (nome[i] != ' ')

	{

		i++;

	}

	i++;

	pnome = &nome[i];

	return pnome;

}

int main()

{

	char nomeCompleto[80];

	char *p;

	puts("Entre com o seu nome e um sobrenome.");

	fgets(nomeCompleto, 80, stdin);

	p = acheSobrenome(nomeCompleto);
```

```
	puts(p);
	return 0;
}
```

Exercício 5

```
#include<stdlib.h>
#include<stdio.h>
int main()
{
	int i, n, *pvetor;
	float media;
	scanf("%i", &n);
	pvetor = (int *)malloc(n * sizeof(int));
	if (!pvetor)
	{
		puts("Sem memoria.");
		return 1;
	}
	for (i =0; i < n; i++)
	{
		scanf("%i", &pvetor[i]);
	}
	media = 0.0;
	for (i = 0; i <  n; i++)
	{
		media += pvetor[i];
	}
	media = media / i;
	printf("%f\n", media);
	free(pvetor);
```

```
	return 0;
}
```

Exercício 6

```
#include<stdlib.h>
#include<stdio.h>

int main()
{
	int i, n, *pvetor;
	float media;
	scanf("%i", &n);
	pvetor = (int *)malloc(n * sizeof(int));
	if (!pvetor)
	{
		puts("Sem memoria.");
		return 1;
	}
	for (i =0; i < n; i++)
	{
		scanf("%i", &pvetor[i]);
	}
	media = 0.0;
	for (i = 0; i <  n; i++)
	{
		media += pvetor[i];
	}
	media = media/i;
```

```
	printf("%f\n", media);
	free(pvetor);
	return 0;
}
```

Exercício 7

```
#include<stdlib.h>
#include<stdio.h>

int main()
{
	int i, n, *pvetor;
	float media, aux, aux1, j = 0;
	scanf("%i", &n);
	pvetor = (int *)malloc(n * sizeof(int));
	if (!pvetor)
	{
		puts("Sem memoria.");
		return 1;
	}

	for (i =0; i < n; i++)
	{
		scanf("%i", &pvetor[i]);
	}
	media = 0.0;
	for (i = 0; i <  n; i++)
	{
		media += pvetor[i];
```

```
        }
        media = media /i;
        printf("%f\n", media);
        for (i = 0; i <  n; i++)
        {
                if(pvetor[i] > media)
                {
                        aux++;
                }
        }
        printf("%.0f numeros sao maiors que a media sendo as maiores notas:\n", aux);
        for (i = 0; i <  n; i++)
        {
                if(pvetor[i] > media)
                {
                        printf("%i\n", pvetor[i]);
                }
        }
        free(pvetor);
        return 0;
}
```

Exercício 8

```
#include<stdlib.h>

#include<stdio.h>


int main()

{

	int i, n, *pvetor, aux, j = 0;

	float media ,aux1;

	scanf("%i", &n);

	pvetor = (int *)malloc(n * sizeof(int));

	if (!pvetor)

	{

		puts("Sem memoria.");

		return 1;

	}


	for (i =0; i < n; i++)

	{

		scanf("%i", &pvetor[i]);

	}

	printf("Vetor nao ordenado:\n");
```

```
	for (i = 0; i <  n; i++)
	{
		printf("%i\n", pvetor[i]);
	}
	for (i = 0; i <  n; i++)
	{
		for (j = 0; j <  n - 1 - i; j++)
		{
			if(pvetor[j]  > pvetor[j +1])
			{
				aux = pvetor[j];
				pvetor[j] = pvetor[j+ 1];
				pvetor[j + 1] = aux;
			}
		}
	}
	printf("Vetor ordenado:\n");
	for (i = 0; i <  n; i++)
	{
		printf("%i\n", pvetor[i]);
	}
	free(pvetor);
	return 0;
}
```

Exercício 9

```
#include<stdlib.h>

#include<stdio.h>


int main()

{

	int i, aux, j = 0, n;

	double *pvetor;

	do

	{

		n += 100;

		pvetor = (double *)malloc(n * sizeof(int));

		if(!pvetor)

		{

			j = 1;

		}

		else

		{

			free(pvetor);
```

```
            }
        }while(j != 1);
        printf("%i e o maximo de memoria\n", n);

        free(pvetor);
        return 0;
}
```

Exercício 10

A função fgets pega o nome e sobrenome e armazena na String *nome*, com o nome armazenado a string é passada para o procedimento *mistério* por referencia para o ponteiro *n que vai receber o endereço de memoria que a string *nome* aponta e então ele procura onde inicia o sobrenome e o printa na tela. Por fim passa por referencia essa posição na memoria para a string *nome* e termina o programa.

Comparando com o exercício 4, vemos que o exercício 7 retorna a posição de memoria por referencia pois ele recebe o nome por referencia e dentro do procedimento ele trabalha no próprio endereço de memoria logo tudo que ele faz com a variável *n ira alterar oque a string *nome* ira mostrar já que ambas a pontam pro mesmo segmento de memoria. Já no exercício 4 a string *nomeCompleto* não é passada por referencia logo tudo oque os ponteiros de *acheSobrenome* alteram dentro do procedimento antes de ser retornado para main não afetam a variável *nomeCompleto* na main, Somente depois de retornado a main a variável *nomeCompleto* será atualizada com oque aconteceu no procedimento.

```
#include<stdlib.h>
#include<stdio.h>

void misterio (char *n);
int main()
{
        char nome[41];
        fgets(nome, 41, stdin);
```

```
	misterio(nome);
	return 0;
}
void misterio (char *n)
{
	while (*n != ' ')
	n++;
	n++;
	puts(n);}
```

Exercício 11

```
#include<stdio.h>
#include<stdlib.h>

void Converter (int numeroBase10, int numeroBase2[]);

int main(int argc, char *argv[])
{
	int nb10, nb2[32], i, teste = 1;
	while(1)
	{
		scanf("%i", &nb10);
		if(nb10 < 0) break;
		Converter(nb10, nb2);
		printf("Teste %i\n", teste++);
		printf("%i\n", nb10);
		for(i = 0; i < 32; i++)
		{
			printf("%i", nb2[i]);
```

```
                }
                printf("\n\n");
        }
        return 0;
}

void Converter(int numeroBase10, int numeroBase2[])
{
        int i = 0, resto[32], j, k = 0, p;
        do
        {
                resto[i] = numeroBase10 % 2 ;
                numeroBase10 = numeroBase10 / 2;
                i++;
        }while(numeroBase10 >= 2);
        resto[i] = numeroBase10;
        i++;
        p = i - 1;
        for(j = 0; j < 32 - i;j++)
        {
                numeroBase2[j] = 0;
                k++;
        }
        for(j = k; j < 32;j++)
        {
                numeroBase2[j] = resto[p];
                p--;
```

```
        }
    }
```