Lista 1 de Recursividade

Aluno: Gabriel Lima Scheffler RA: 2677180

Ex1:

```
#include <stdio.h>
#include <stdlib.h>
int invert(int n){
   if (n > 0) {
      //aqui ele printa o ultimo numero como % 10
      printf("%d", n % 10);
      // aqui ele puxa o penultimo numero apartir da divisao por 10
      invert(n / 10); }}
int main() {
// numero de variavel escolhida
int numero = 123456789;
//Chamada da função recursiva associando seu parametro à variavel numero na main
invert(numero);
return 0;
}
```

Neste exercicio eu crio uma função recursiva que pega um numero inteiro como parametro inicial, depois uma verificação para que nao haja numeros menores que 0 e printa o ultimo numero sendo ele por % 10 que puxa o ultimo algarismo de um numero, depois eu realizo a recursividade dividindo o numero por 10 colocando o ultimo algarismo nas casas decimais e sem contagem e depois eu continuo na função de recursividade

Ex2:

```
#include <stdio.h>

#include <stdlib.h>

//Função Recursiva

int Soma(int n[],int t){

  //verficação de quando o tamanho do vetor for igual a 0 para ele retornar

  if (t == 0) return 0;

  else{

  //return da recursividade com o proximo numero

    return n[t - 1] + Soma(n, t - 1);

  }

}

int main() {

  //declaração do vetor

int vetor[] = {1,2,3};

  //Calculo em t para descobrir o tamanho do vetor para ser posteriormente inserido na função

int t = sizeof(vetor) / sizeof(vetor[0]);

//print da soma do vetor com a chamada de função

printf("A soma: %d\n",Soma(vetor,t));

return 0;

}
```

Neste exercicio eu criei uma função recursiva que verificava quando um numero era zero, ela deveria retornar o resultado e no else ela diminuia o tamanho para jogar para a outra variavel no vetor, e assim continuar com as somas, no main eu criei o vetor e criei tambem uma variavel que conta o tamanho do vetor para ser jogado nos parametro da função

Ex3:

```
#include <stdio.h>

#include <stdlib.h>

//Função Recursiva

int Soma(int n){

  //Verificação de quando o numero for zero ele retorne o resultado

  if (n == 0) return 0;

  else {

    // return da recursividade com a soma de por exemplo 10 + 9, no caso a soma do numero pela subtração do mesmo

    return n + Soma(n - 1);

  }

}

int main() {

  // Declaração de variavel

int numero = 10;

// Print com a chamada da função recursiva

printf("A soma = %d\n",Soma(numero));

return 0;

}
```

Aqui neste exercicio eu crio uma função que pega como parametro um numero e ele deve fazer a soma de todos ate 1, segue a mesma ideia dos outros, mas pegando o numero e puxando dentro da função a propria com a variavel apenas subtraindo por 1

Ex4:

```
#include <stdio.h>
#include <stdlib.h>
//Função Recursiva
int Potencia(int n, int k){
   //Verificação de quando o numero for zero ele retorne o resultado vezes 1
   if (k == 0) return 1;
   else{
      // multiplica n por n mas subtrai k - 1 continuando a recursividade
      return n * Potencia(n, k - 1);
   }
}
int main() {
   //declara as variaveis n e k
   int n,k;
   //input de n
   printf("Digite um numero para n : ");
   scanf("%d",&n);
   //input de k
   printf("Digite um numero para k : ");
   scanf("%d",&k);
   //atrela a variavel result o resultado da função potencia
   int result = Potencia(n,k);
   //print de result
   printf("A potencia = %d\n", result);
return 0;
}
```

Neste Exercicio eu fiz a potenciação de um numero apenas pegando o k e subtraindo por 1 na recursividade

Ex5:

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
void invert(int n[], int i, int f) {
  //Verificação para quando o inicio e final do vetor forem iguais ele retorne
  if (i >= f) return;
  else {
    //Aqui ele troca as posiçoes dos vetores
    int troca = n[i];
    n[i] = n[f];
    n[f] = troca;
    // aqui na recursividade ele diminui a primeira e ultima para na proxima ele pegar a segunda e penultima
    invert(n, i + 1, f - 1);}}
int main() {
  //declara o vetor
  int vet[100];
  //insere valores nos vetores
  for (int i = 0; i < 100; i++) {
    vet[i] = i + 1;
  }
  // aciona a função
  invert(vet, 0, 99);
  // for para mostrar o resultado invertido
  for (int i = 0; i < 100; i++) {
    printf("o vetor invertido = %d\n", vet[i]);
  }
```

```
return 0;}
```

Neste exercicio eu crio um vetor de 100 posiçoes e insiro numeros de 1 a 100 nele, depois eu na função defino 3 parametros sendo eles o vetor, o inicio e fim e depois eu apenas inverto a primeira posição pela ultima e na recursividade ele subtrai para pegar a segunda e penultima, depois na main eu apenas printo o resultado

Ex6:

```
#include <stdio.h>

#include <stdlib.h>

// Função Recursiva

int mdc(int x, int y) {

  //Verificação para quando y for zero ele retorne x, o maximo divisor comum para zero

  if (y == 0 ) return x;

  else {

    // recursao para ele dividir sempre pelo resto e assim quando zera, ele retorna o
mdc

    return mdc(y, x%y);}}

int main() {

  // atribuição de variaveis

  int x = 1071;

  int y = 462;

  // print com atribuição de variaveis na função recursiva

  printf("Resultado : %d\n",mdc(x,y) );

  return 0;}
```

Nesta função é utilizada a forma como passada no exercicio sendo feita a divisao dos numeros sempre pelo resto ate chegar a zero, assim é possivel encontrar o mdc entre dois numeros

Ex7

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
int contador(int x, int y) {
   //Verificação para quando y for zero ele retorne 0 pois nao ha nada para contar
   if (y == 0 ) return 0;
   else{
      // eu criei a variavel que le o ultimo numero
      int num = y % 10;
      // aqui eu usei a mesma logica do EX.1 que eu comparo e removo o numero
      return (num == x) + contador(x, y / 10);}}
int main() {
   // declaração de variaveis
   int x,y;
   // input para X
   printf("Digite um numero para x  :");
   scanf("%d",&x);
   // input para y
   printf("Digite um numero para y : ");
   scanf("%d",&y);
   // print com chamada de função recursiva
   printf("Resultado : %d\n", contador(x,y));
   return 0;}
```

Nesta função eu criei um contador que analisa a ultima linha do numero inserido y e compara com x e depois remove a ultima dividindo por 10 deixando a ultima casa fora do grupo dos inteiros, e assim eu comparo

Ex8

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
int Multip_Rec(int x,int y)  {
   //Verificação para quando y for zero ele retorne 0 pois nao ha nada para multiplicar
   if (y == 0 ) return 0;
   else{
   //Retorna a recursividade com a soma de x com x e subtrai o y a cada recursividade
      return x + Multip_Rec(x,y-1);}}
int main() {
   // declara as variaveis
   int x = 20;
   int y = 3;
   // print do resultado chamando a função recursiva
   printf("Resultado : %d\n", Multip_Rec(x,y));
   return 0;}
```

Nesta função ela pega o x e soma com o mesmo e enquanto isso subtrai o y ate chegar a 0, a ideia é fazer as somas enquanto o y nao chega a zero, assim printando o resultado no final, na main

Ex 9-

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
int par(int x, int i)  {
   //Verificação para quando x for zero ele retorne a função
   if (i == x ) return 0;
   // if para ver o numero par e printar
      if(i % 2 == 0) printf("%d", i);
      // soma 1 para printar todos os pares de 0 a N
         return par(x,i + 1);}
int main() {
   // declara as variaveis
   int x;
   //input para x
   printf("Digite um numero para x : ");
   scanf("%d",&x);
   // print do resultado chamando a função recursiva com i == 0
   par(x,0);
   return 0;}
```

Nesta função eu fiz os numeros irem de 0 a n mostrando os numeros pares por recursao, e atribui x como o numero n e a variavel i como o contador ate x e verficaçoes de numeros pares na função, ou seja, i deve ser igual a x para finalizar a recursao e enquanto nao seja ele soma 1 a i e verifica se eh par ate o x

Ex 10

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
int par(int x)  {
   //Verificação para quando x for zero ele retorne a função
   if (x == 0 ) return 0;
   // if para ver o numero par e printar
      if(x % 2 == 0) printf("%d", x);
      // subtrai 1 para printar todos os pares
         return par(x - 1);}
int main() {
   // declara as variaveis
   int x;
   //input para x
   printf("Digite um numero para x : ");
   scanf("%d",&x);
   // print do resultado chamando a função recursiva
   par(x);
   return 0;}
```

função recursiva para descobrir se o numero é par e printar e diminuir 1 sempre na recursao, assim printando na main o resultado correto

Ex 11 –

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
int par(int x)  {
   //Verificação para quando x for zero ele retorne a função
   if (x == 0 ) return 0;
   // if para ver o numero par e printar
      if(x % 2 == 0) printf("%d", x);
      // subtrai 1 para printar todos os pares
         return par(x - 1);}
int main() {
   // declara as variaveis
   int x;
   //input para x
   printf("Digite um numero para x : ");
   scanf("%d",&x);
   // print do resultado chamando a função recursiva
   par(x);
   return 0;}
```

função recursiva para descobrir se o numero é par e printar e diminuir 1 sempre na recursao, assim printando na main o resultado correto

Ex 12 –

```
#include <stdio.h>
#include <stdlib.h>
// Função Recursiva
unsigned long long cal(int num) {
  //if para verificar se o numero é maior ou igual que 0
  if (num<=0) return 1;
  // multiplica o num pelo num recursivo - 2 para pegar os impares
  return num * cal(num- 2);}
int main() {
  // declara a variavel
  int numero;
  //input
  printf("Digite um numero para realizar o Fatorial Duplo : ");
  scanf("%d", &numero);
  // chamada da função
  printf("Fatorial Duplo de %d = %llu\n", numero, cal(numero));
  return 0;}
```

Nesta função eu pego um numero e na  função cal eu pego um numero e multiplico ele por ele mesmo recursivo menos 2, sendo assim pegando os numeros impares