Exercícios sobre Passagem por Referência

1. Altere as chamadas de funções abaixo de forma que a função chamada não tenha retorno e, ao mesmo tempo, a variável que antes receberia o valor retornado possa ser alterado diretamente na função chamada.

ano = lerDado(); lerDado(&ano);

a = recebendoUmValor(); recebendoUmValor(&a);

idade = calcularIdade(ano); calcularIdade(ano, &idade);

b = calcularAlgo(a); calcularAlgo(a, &b);

c = somar(a,b); somar(a, b, &c);

c = fazAlgo(a,b); fazAlgo(a, b, &c);

d = getRA(); getRA(&d);

e = getValor(); getValor(&e);

1. Altere as funções abaixo, de forma que não retornem mais valores e, ao mesmo tempo, atribuam o valor definido diretamente no endereço de memória recebido.

|  |  |
| --- | --- |
| int retornandoUmNumero(){  int a;  a = 40;  return a;  } | int retornaValorAleatorio(){  int valor;  srand(time(NULL));  valor = rand()%10;  return valor;  } |
| int calcularIdade(int ano){  int idade;  idade = 2019-ano;  return idade;  } | int calcularAno(int idade){  int ano;  ano = 2019-idade;  return ano;  } |
| int somar(int a, int b){  int res;  res = a+b;  return res;  } | int subtrair(int a, int b){  int res;  res = a-b;  return res; |

|  |  |
| --- | --- |
| void retornandoUmNumero(int \*a){  \*a = 40;  } | void retornaValorAleatorio(int \*valor){  srand(time(NULL));  \*valor = rand()%10;  } |
| void calcularIdade(int ano, int \*idade){  \*idade = 2019-ano;  } | void calcularAno(int idade, int \*ano){  \*ano = 2019-idade;  } |
| void somar(int a, int b, int \*res){  \*res = a+b;  } | void subtrair(int a, int b, int \*res){  \*res = a-b;  } |

1. Altere os códigos abaixo de forma que eles não retornem mais valores. No lugar, use ponteiros.

|  |  |
| --- | --- |
| int main(int argc, char\*\* argv) {  int valor;  int quadrado;  printf(“\nDigite um valor:”);  scanf("%d",&valor);    calcularQuadrado(valor, &quadrado);  printf(“\n%d ao quadrado = %d”, valor, quadrado)  return 0;  } | void calcularQuadrado(int valor, int \*quadrado){  \*quadrado = valor \* valor;  } |
| int main(int argc, char\*\* argv) {  float celsius;  float kelvin;  printf("\nDigite a temperatura em Celsius:");  scanf("%f",&celsius);    converterCparaK(celsius, &kelvin);  printf("\n%.2f Celsius = %.2f Kelvin",celsius,kelvin);  return 0;  } | void converterCparaK(float celsius, float \*kelvin){  \*kelvin = c + 273.15f;  } |

|  |  |
| --- | --- |
| int main(int argc, char\*\* argv) {  float celsius;  float kelvin;  printf("\nDigite a temperatura em Celsius:");  scanf("%f",&celsius);    converterCparaK(celsius, &kelvin);  printf("\n%.2f Celsius = %.2f Kelvin",celsius,kelvin);  return 0;  } | void converterCparaK(float celsius, float \*kelvin){  \*kelvin = c + 273.15f;  } |

|  |  |
| --- | --- |
| int main(int argc, char\*\* argv){  int valor1, valor2;  int resultado;  printf(“\nDigite um valor: “);  scanf("%d",&valor1);  printf("\nDigite outro valor:");  scanf("%d",&valor2);  multiplicar(valor1, valor2, &resultado);  printf("\n%d x %d = %d ",valor1, valor2, resultado);  return 0;  } | void multiplicar(int valor1, int valor2, int \*resultado){  \*resultado = a\*b;  } |

1. Para cada código do exercício 3, preencha a tabela de endereços, supondo que a primeira variável seja sempre alocada no endereço 55FF20.

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| valor | 0055FF20 | 3 |
| quadrado | 0055FF1C | 9 |
| ~~val~~ | ~~0055FF18~~ | ~~3~~ |
| ~~quad~~ | ~~0055FF14~~ | ~~9~~ |

Tabela algoritmo antigo

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| valor | 0055FF20 | 3 |
| quadrado | 0055FF1C | 9 |
| ~~valor~~ | ~~0055FF18~~ | ~~3~~ |
| ~~quadrado~~ | ~~0055FF14~~ | ~~0055FF1C~~ |

Tabela algoritmo corrigido

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| celsius | 0055FF20 | 0 |
| kelvin | 0055FF1C | 273.15 |
| ~~c~~ | ~~0055FF18~~ | ~~0~~ |
| ~~k~~ | ~~0055FF14~~ | ~~273.15~~ |

Tabela algoritmo antigo

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| celsius | 0055FF20 | 0 |
| kelvin | 0055FF1C | 273.15 |
| ~~celsius~~ | ~~0055FF18~~ | ~~0~~ |
| ~~kelvin~~ | ~~0055FF14~~ | ~~0055FF1C~~ |

Tabela algoritmo corrigido

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| celsius | 0055FF20 | 0 |
| kelvin | 0055FF1C | 273.15 |
| ~~c~~ | ~~0055FF18~~ | ~~0~~ |
| ~~k~~ | ~~0055FF14~~ | ~~273.15~~ |

Tabela algoritmo antigo

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| celsius | 0055FF20 | 0 |
| kelvin | 0055FF1C | 273.15 |
| ~~celsius~~ | ~~0055FF18~~ | ~~0~~ |
| ~~kelvin~~ | ~~0055FF14~~ | ~~0055FF1C~~ |

Tabela algoritmo corrigido

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| valor1 | 0055FF20 | 2 |
| valor2 | 0055FF1C | 5 |
| resultado | 0055FF18 | 10 |
| ~~a~~ | ~~0055FF14~~ | ~~2~~ |
| ~~b~~ | ~~0055FF10~~ | ~~5~~ |
| ~~res~~ | ~~0055FF0C~~ | ~~10~~ |

Tabela algoritmo antigo

|  |  |  |
| --- | --- | --- |
| Identificador | Endereço | Valor |
| valor1 | 0055FF20 | 2 |
| valor2 | 0055FF1C | 5 |
| resultado | 0055FF18 | 10 |
| ~~valor1~~ | ~~0055FF14~~ | ~~2~~ |
| ~~valor2~~ | ~~0055FF10~~ | ~~5~~ |
| ~~resultado~~ | ~~0055FF0C~~ | ~~0055FF18~~ |

Tabela algoritmo corrigido