|  |  |  |  |
| --- | --- | --- | --- |
| 1. Dado o código abaixo, qual o resultado a ser apresentado no console? (1,0)  |  |  | | --- | --- | | **int oquefaz(int a){**  **if(a==1)**  **return a + a \* a;**  **else{**  **return oquefaz(a-1);**  **}**  **}** | **int main(int argc, char \*argv[]) {**  **int a = oquefaz(6);**  **printf("%d",a);**  **return 0;**  **}** |   **2.** Considere que, no próximo código, a função main chama a função func, passando 3 como argumento. O que será impresso na tela? Desenhe a execução. (1,5)  **void func (int n){**  **if (n == 0)**  **printf(“fim”);**  **else**  **{**  **func(n-1);**  **printf(“%d”, n);**  **func(n-1);**  **}**  **}**  3. Faça um algoritmo que peça um número inteiro e calcule a quinta potência deste valor usando recursividade. Use a estrutura básica de uma função recursiva que realiza cálculos e retorna um valor ensinada em aula (1,5). | 4. Transforme o código a seguir em uma função recursiva (2,0):  **void fazAlgo(){**  **for(int i=1;i<4;i++){**  **if(i%2==0)**  **printf(“\n%d é par”,i);**  **else**  **printf(“\n%d é ímpar”,i);**  **}**  **}**  5. Transforme o código a seguir em uma função não recursiva (2,0):  **void fazAlgo(){**  **int a;**  **printf("\nDigite algo:");**  **scanf("%d",&a);**  **switch(a){**  **case 1:**  **printf("\nVocê digitou 1");**  **break;**  **case 2:**  **printf("\nVocê digitou 2");**  **break;**  **}**  **if(a!=0){**  **fazAlgo();**  **}**  **}** |

**Respostas:**

1 - O código fornecido implementa uma função recursiva chamada oquefaz, que recebe um argumento a. Se a for igual a 1, a função retorna a + a \* a. Caso contrário, a função chama a si mesma com o argumento a-1. No main, a função oquefaz é chamada com o argumento 6, e o resultado é armazenado na variável a. Em seguida, o valor de a é impresso no console usando printf.

A execução da função oquefaz(6) envolve chamadas recursivas até atingir o caso base, que é oquefaz(1). Neste caso base, o valor retornado é 1 + 1 \* 1, que é igual a 2. Portanto, o resultado impresso no console será 2.

2 - #include <cstdio>

void func(int n) {

if (n == 0)

printf("fim");

else {

func(n - 1); // Chama a função com n-1

printf("%d", n); // Imprime o valor de n

func(n - 1); // Chama a função novamente com n-1

}

}

int main() {

func(3); // Chama a função func com o argumento 3

return 0;

}

func(3) chama func(2) (primeira chamada recursiva).

func(2) chama func(1) (segunda chamada recursiva).

func(1) chama func(0) (terceira chamada recursiva).

Em func(0), a condição if (n == 0) é verdadeira, então imprime "fim".

Agora, voltando para a chamada func(1), após a chamada recursiva func(0), imprime "1". Então, a chamada func(1) imprime "fim1".

Retornando para a chamada func(2), após as chamadas recursivas, imprime "2". Então, a chamada func(2) imprime "fim1fim2".

Finalmente, retornando para a chamada func(3), após as chamadas recursivas, imprime "3". Então, a chamada func(3) imprime "fim1fim2fim1fim3fim1fim2fim1fim".

Desenho das chamadas para a entrada func(3):

func(3)

|-- func(2)

| |-- func(1)

| | |-- func(0) -> "fim"

| | |-- print("1") -> "fim1"

| | |-- func(0) -> "fim"

| |-- print("2") -> "fim1fim2"

| |-- func(1)

| |-- func(0) -> "fim"

| |-- print("1") -> "fim1fim2fim1"

| |-- func(0) -> "fim"

|-- print("3") -> "fim1fim2fim1fim3"

|-- func(2)

|-- func(1)

| |-- func(0) -> "fim"

| |-- print("1") -> "fim1fim2fim1fim3fim1"

| |-- func(0) -> "fim"

|-- print("2") -> "fim1fim2fim1fim3fim1fim2"

|-- func(1)

|-- func(0) -> "fim"

|-- print("1") -> "fim1fim2fim1fim3fim1fim2fim1"

|-- func(0) -> "fim"

3 - #include <iostream>

// Função recursiva para calcular a quinta potência

int quintaPotencia(int base, int expoente) {

// Caso base: expoente é 1, retorna a base

if (expoente == 1) {

return base;

} else {

// Chamada recursiva para reduzir o expoente

return base \* quintaPotencia(base, expoente - 1);

}

}

int main() {

int numero;

// Solicita ao usuário que insira um número inteiro

std::cout << "Digite um numero inteiro: ";

std::cin >> numero;

// Calcula a quinta potência usando a função recursiva

int resultado = quintaPotencia(numero, 5);

// Exibe o resultado

std::cout << "A quinta potencia de " << numero << " eh: " << resultado << std::endl;

return 0;

}

Neste código, a função quintaPotencia é recursiva e calcula a potência da base elevada ao expoente. O caso base é quando o expoente é 1, onde a função retorna a base. Caso contrário, a função faz uma chamada recursiva com um expoente reduzido em 1. O programa principal pede ao usuário um número inteiro, calcula a quinta potência usando a função recursiva e exibe o resultado.

4 - #include <iostream>

void fazAlgoRecursivo(int i, int n) {

if (i <= n) {

if (i % 2 == 0)

std::cout << "\n" << i << " é par";

else

std::cout << "\n" << i << " é ímpar";

fazAlgoRecursivo(i + 1, n);

}

}

int main() {

int n;

// Solicita ao usuário fornecer um valor para 'n'

std::cout << "Digite um valor para n: ";

std::cin >> n;

// Chama a função recursiva

fazAlgoRecursivo(1, n);

return 0;

}

Esta versão do código solicita ao usuário fornecer um valor para 'n', permitindo que o programa seja mais flexível. Além disso, mantém a estrutura recursiva da função fazAlgoRecursivo.

5 - #include <cstdio>

void fazAlgo() {

int a;

do {

printf("\nDigite algo:");

scanf("%d", &a);

switch (a) {

case 1:

printf("\nVocê digitou 1");

break;

case 2:

printf("\nVocê digitou 2");

break;

}

} while (a != 0);

}

int main() {

fazAlgo();

return 0;

}

Neste código, o loop do-while permite que o programa continue pedindo ao usuário para digitar algo até que o valor digitado seja 0. Isso substitui a chamada recursiva da função fazAlgo.