Lista 2

Aluno: Nathann Zini dos Reis

Matrícula: 19.2.4007

Q.1 - Explique o funcionamento do mecanismo chamado “call-back function”.

Resumidamente, o mecanismo call-back function é responsável por fazer uma chamada a uma função/um executável que é passado por argumento para outro código, que, em algum momento, vai ser executado "call back".

Q.2 - Explique o funcionamento do código contido no arquivo “main.cpp”.

O código contido no main.cpp cria uma tela, passando os dados referentes à estrutura da janela, como tamanho de pixel de altura e largura, cor do background e etc.

Q.3 - O que o código contido no arquivo “main1.cpp” faz?

O código do main1.cpp escreve na tela "Hello World", depois, se o caractere contido em c for diferente de 's', ele faz a leitura de um caractere para c, imprime o valor, em inteiro, da subtração dos valores de 'A' e 'a' e escreve na tela o caractere digitado pelo usuário e pausa o programa.

Q.4 - Como este código funciona?

O código do main1.cpp escreve na tela "Hello World", depois, se o caractere contido em c for diferente de 's', ele faz a leitura de um caractere para c, imprime o valor, em inteiro, da subtração dos valores de 'A' e 'a' e escreve na tela o caractere digitado pelo usuário e pausa o programa.

Q.5 - Por que é necessário utilizar as instruções de “type casting”?

É necessário pois está fazendo atribuição de valores de tipo diferente da variável.

Q.6 - O que significa “0xFF” no código “main2.cpp”?

é referente à quantidade de caracteres presente na tabela ASCII que é 255.

Q.7 - O que o primeiro loop do código “main2.cpp” faz e como ele funciona?

Percorre o array de char e enquanto não for o final do array, representado por '\0' e imprime cada caractere na tela.

Q.8 - O que o segundo loop do código “main2.cpp” faz e como ele funciona?

O segundo loop percorre um array de inteiros enquanto o conteúdo do array for diferente de 9, e imprime cada uma dos conteúdos das posições, seguidos por seus endereços de memória. E., no final, imprime o tamanho em bits do inteiro.

Q.9 - O que o terceiro loop do código “main2.cpp” faz? Por que ele não funciona?

Em teoria, o terceiro loop escreveria na tela todos os char contidos na variável pChar seguidos de seus endereços na memória, porém não roda corretamente o código pois a atribuição feita anteriormente, forçando pela type casting o vetor de int dentro do pChar, não é correto ser feito.

Q.10 - O que pode ser concluído sobre a aritmética de ponteiros?

que é possível apontar para a porção da memória que guarda o conteúdo da variável. Isso é bom pois, quando preciso, pode passar a referência da memória e fazer alterações em diferetnes módulos do código.

Q.11- O código possui duas diretivas de pré-compilação que fazem com que o código
“main3.cpp” gere erros. Compile e execute o código com a diretiva ERRO1 ativada e
desativada e, então, Baseado na saída impressa na tela, explique o que o trecho

de
código de linha 27 a 32 faz

No trecho mencionado na pergunta, o código atribui à variável do tipo unsigned int um endereço de memória. Depois é então impresso na tela o endereço de memória da variável, depois o endereço que está guardada na variável, e por fim é impresso o inteiro conteúdo daquele endereço que está guardado na variável.

Q.12- O código possui duas diretivas de pré-compilação que fazem com que o código
“main3.cpp” gere erros. Compile e execute o código com a diretiva ERRO2 ativada e
desativada e, então, baseado na saída impressa na tela, explique o funcionamento do
trecho de código entre as linhas 37 e 57.

No Trecho mencionado na pergunta, o código, similarmente ao que faz na diretiva ERRO1, ele imprime o endereço de memória da variável e depois o endereço guardado dentro nela, que a principio é nulo, logo é validado se aquele endereço é valido. E como é nulo então é FALSO, em seguida é atribuído um endereço de memória àquela variável que então é novamente validada e agora é TRUE.