1. Explique o que é um ponteiro e qual a sua função principal em programação.

Ponteiros são variáveis que referenciam o endereço de memória de outra variável do mesmo tipo. Elas podem fornecer acesso em nível mais baixo a variáveis, permitindo acessar uma única variável em métodos e procedimentos diferentes, economizando memória.

2. Quais são os dois operadores principais utilizados com ponteiros em C++? Descreva a função de cada um deles (o que ele "resulta" ou "acessa").

   * (asterisco) = Utilizada para criar e acessar o valor de uma variável do tipo ponteiro.
   & (e comercial) = utilizada para buscar/acessar o endereço de memória variável que estiver após o símbolo.

3. Explique a importância de inicializar ponteiros durante a declaração e qual valor é frequentemente usado para ponteiros não atribuídos.

   Eles devem ser inicializados pois, ao tentar acessar algum valor de um ponteiro nulo ocorre um erro de memória, referenciando o valor da própria variável. Ponteiros não atribuidos podem possuir o valor nullptr.

4. Como se acessa o valor da variável que um ponteiro aponta?

   Utilizando o asterisco.

5. Como se atribui um valor a uma variável através de um ponteiro?

   Utilizando *ptr = 10;

6. Como se inicializa um ponteiro com o endereço de uma variável (com exceção para arrays)?

   int x = 10, int *ptrX = &x;

7. Diferencie a passagem de parâmetros por valor e a passagem por referência em funções, explicando como os ponteiros são usados para implementar a passagem por referência.

   A passagem por referência, não cria novas variáveis do mesmo tipo e significado para a mesma utilidade, ou seja, em uma função para calcular a idade, poderia passar meu ano de nascimento como um big int, porém, ao utilizar ponteiro, posso economizar espaço na memória, referênciando o big int que já existe na main.

8. Qual a vantagem de passar structs por referência para uma função?

   Utilizar os valores de forma “global”, evitando cópias.

9. Descreva como um vetor é passado para uma função utilizando ponteiro. Explique como funciona a memória alocada.

   De forma resumida, passar um vetor como ponteiro significa passar o endereço de memória do índice 0 do vetor.

10. Quais são as duas formas de acessar as informações de uma struct quando se utiliza um ponteiro para ela?

    (*strUser).username
    ou
    strUser->username

11. Explique a diferença entre alocação automática e alocação dinâmica.
    A alocação dinâmica é feita, geralmente, pelo programador com o objetivo de alocar a quantidade de memória exata para a função daquele programa, com o objetivo de fornecer mais performance na execução. Já a alocação automática é feita pelo compilador, que cria e deleta as variáveis, mas pode alocar mais espaço do que realmente necessário.

12. Liste os benefícios da alocação dinâmica de memória, incluindo como ela impacta o uso de memória e a persistência dos dados alocados.

    Menor desperdício de memória, quantidade alocada sob demanda, alocar memória durante a execução.

13. Quais são os dois operadores em C++ utilizados para o sistema de alocação dinâmica? Explique a função de cada operador.
    new – serve para alocar um novo espaço em memória
    delete – deleta um espaço em memória
14. O que acontece com o espaço de memória alocado dinamicamente se não for explicitamente liberado?

    Pode fornecer um aumento excessivo no uso de memória, trazendo lentidão no computador que executa o programa.

15. Para quais tipos de variáveis não vale a pena usar alocação dinâmica?
    Tipos mais simples como int, char, float e string.
16. Quais são os propósitos principais para utilizar a alocação dinâmica de memória?
    Alocar vetores dinâmicos e alocar variáveis de bloco.
17. Explique como o new[] e delete[] são utilizados para alocar e liberar vetores dinâmicos.
    Dentro dos colchetes, será inserido o valor da matriz e através disso, será alocado o valor para toda a matriz.

18. Descreva como a alocação dinâmica de struct pode ser realizada, tanto para uma única struct quanto para um array de struct.

A principal diferença está na utilização de colchetes para a alocação e para a deleção dos valores na memória.