1. Каким образом можно задавать указатель на массив?

Указатель на массив можно объявить следующим образом:

int arr[5] = {1, 2, 3, 4, 5};

int\* ptr = arr; // ptr указывает на первый элемент массива

Также можно явно указать, что это указатель на массив:

int (\*ptr)[5] = &arr; // ptr указывает на весь массив

2. Каким образом можно задавать указатель на функцию?

Указатель на функцию задается следующим образом:

int func(int a) { return a \* 2; }

int (\*ptrFunc)(int) = func; // ptrFunc - указатель на функцию

Вызов:

int result = ptrFunc(10);

3. Для чего предназначена функция malloc()?

Она выделяет динамическую память в куче и возвращает указатель на первый байт выделенной памяти. Используется в C/C++.

int\* ptr = (int\*)malloc(5 \* sizeof(int)); // Выделяем память для 5 int

4. Что делает функция free() и в какой библиотеке она определена?

Функция free() освобождает динамически выделенную память. Определена в <stdlib.h>.

free(ptr);

5. Как можно использовать указатели при работе с массивами?

Указатели позволяют обходить массив с использованием арифметики указателей:

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

int\* ptr = arr;

for (int i = 0; i < 3; i++) {

printf("%d ", \*(ptr + i));

}

6. Как можно обращаться к элементам массива через указатель?

int arr[] = {10, 20, 30};

int\* ptr = arr;

printf("%d\n", \*ptr); // 10

printf("%d\n", \*(ptr+1)); // 20

7. Как при помощи указателя можно передать в функцию значение переменной, массив?

Передача переменной:

void modify(int\* x) { \*x = 42; }

int a = 10;

modify(&a);

Передача массива:

void printArray(int\* arr, int size) {

for (int i = 0; i < size; i++) printf("%d ", arr[i]);

}

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

printArray(arr, 3);

8. Как при помощи указателя из функции можно возвратить два и более значений?

Можно передавать указатели на переменные:

void swap(int\* a, int\* b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

9. Что такое операции new и delete? Как при помощи new создать динамический массив?

new выделяет память, delete освобождает:

int\* arr = new int[5]; // Динамический массив на 5 элементов

delete[] arr;

10.Как при помощи new создать двумерный динамический массив?

int\*\* arr = new int\*[rows];

for (int i = 0; i < rows; i++) {

arr[i] = new int[cols];

}

11.Для чего предназначена функция malloc()?

Для выделения динамической памяти.

12. Что делает функция free() и в какой библиотеке она определена?

Освобождает динамическую память, <stdlib.h>.

13. Как выделяется область динамической памяти? Какие функции используются

В C: malloc(), calloc(), realloc(). В C++: new.

14. Как высвобождается область динамической памяти? Какие функции используются?

В C: free(), в C++: delete и delete[].

15. Может ли функция возвращать значение типа указатель?

Да:

int\* createArray(int size) {

return new int[size];

}

16.Могут ли параметрами функции быть указатели на объекты

Да, передача по указателю:

void modifyObject(MyClass\* obj) { obj->field = 10; }

17. Могут ли параметрами функции быть указатели на функции?

void execute(int (\*func)(int), int value) {

printf("%d\n", func(value));

}

18.Как передаются массивы в качестве аргументов функции?

Передаются как указатели:

void printArray(int\* arr, int size);

19.Что такое "висячие" указатели и как этого избежать?

Висячий указатель — это указатель, который указывает на уже освобожденную память. Нужно присваивать nullptr после free()/delete.

20. Из-за чего происходит "утечка" памяти?

Неосвобождение памяти после new или malloc().

21. Типичные ошибки при работе с динамической памятью

- Использование delete вместо delete[]

- Доступ к освобожденной памяти

- Утечки памяти

22. Как проверить, выделена ли память?

int\* ptr = (int\*)malloc(10 \* sizeof(int));

if (!ptr) { printf("Ошибка выделения памяти\n"); }