**Из каких сегментов состоит структура памяти процесса?**

**stack (стек)** - для локальных объектов (в функции), адреса возврата;

**свободное пространство** –

**heap (куча)** - динамическое выделение, берет необходимые объемы памяти из свободного пространства

**.data –** сегмент со статическими данными, переменными

**.text –** место, где хранится сам код программы

**Каким образом связаны встроенные указатели и массивы?**

И указатели, и массивы относятся к низкоуровневым средствам.

Указатель – ссылка на какую-то ячейку памяти, связанную с определенным типом данных.

Имя массива – ссылка на ячейку памяти, содержащюю первую ячейку массива.

Возможные операции:

Int \*ptr = new int[13]; - указатель на память выделенную для массива, состоящего из 13 элементов типа int.

Если задано int v[5]; и int\* ptr = v[0]; То ptr = v; ptr = &v[0]; \*ptr = v[0]

**Почему низкоуровневая работа с памятью небезопасна?**

Нужно постоянно следить за тем, чтобы не было утечек памяти. Чтобы на каждый new в программе имелся корректный delete. Плюс нужно быть внимательным с использованием \* (разыменования), потому что при работе с большой программой из-за пропуска лишь одного символа \* можно потерять много времени на поиски ошибки. Также у низкоуровневых средств не такой большой функционал, поэтому приходится что-то придумывать, тем самым усложняя код

**Что такое ссылка и чем она отличается от указателя?**

Ссылка  - способ обращения к объекту.

Сылка отличается от указателя тем, что

указатель может быть переназначен любое количество раз, в то время как ссылка после привязки не может быть перемещена на другую ячейку памяти (то есть ссылка - это константный указатель);

указатели могут указывать "в никуда" (быть равными null), в то время как ссылка всегда указывает на определенный объект, поэтому ссылки обязательно нужно инициализировать;

у ссылки нельзя получить адрес, а у указателя можно;

не существует арифметики ссылок, в то время как существует арифметика указателей.

**Какими способами можно передать данные в функцию?**

По значению - копирование; int func(int c);

По указателю - нет копирования; int func(int\* c);

По ссылке - автоматически разыменуемый при использовании const - указатель без nullptr. Int func(int &c);

Константная ссылка int func(const int &c)

В константном указателе есть разница: int func(const int\* c) и int func(int\* const c) не одно и то же