

**УВАЖАЕМЫЕ СТУДЕНТЫ!** Законспектируйте в своей рабочей тетради по дисциплине приведенную лекцию (объемом 4-5 страницы), ответьте письменно на контрольные вопросы.

Результаты работы, фотоотчет, предоставить преподавателю на e-mail: [r.bigangel@gmail.com](mailto:r.bigangel@gmail.com) до 06.02.2023.

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: <https://vk.com/daykini>

***ВНИМАНИЕ!!!*** При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

#### Лекция 4

##### Тема: «Понятие программного интерфейса, его назначение»

##### Понятие программного интерфейса и его назначение

Интерфейс — это способ общения пользователя с персональным компьютером, пользователя с прикладными программами и программ между собой. Интерфейс служит для удобства управления программным обеспечением компьютера. Интерфейсы бывают однозадачные и многозадачные, однопользовательские и многопользовательские. Интерфейсы отличаются между собой по удобству управления программным обеспечением, то есть по способу запуска программ. Существуют универсальные интерфейсы, допускающие все способы запуска программ. Универсальные интерфейсы — это псевдобезопасные аналоги не универсальных интерфейсов, предназначенные для сортировки и сравнения на равенство, а также для реализации функциональности, совместно используемой универсальными коллекциями.

Ряд важнейших программ, например все виды DOS, запускаются автоматически при включении компьютера, другие с помощью файлов autoexec.bat или config.sys (различные драйверы). Ряд программ могут при необходимости автоматически загружаться при запуске оболочек Windows.

Графический пользовательский интерфейс. В этом случае пользователь для выполнения нужного действия с помощью мыши выбирает на экране нужный пункт меню или графический символ.

Интерфейс прикладного программирования прикладные программисты используют в своих приложениях обращения к ОС, когда для выполнения тех или иных действий им требуется особый статус, которым обладает только операционная система. В большинстве современных ОС все действия, связанные с управлением аппаратными средствами компьютера, может выполнять только ОС. Помимо этих функций прикладной программист может воспользоваться набором сервисных функций ОС, которые упрощают написание приложений. Функции такого типа реализуют универсальные действия, часто требующиеся в различных приложениях, такие, например, как обработка текстовых строк. Эти функции могли бы быть выполнены и самим приложением, однако гораздо проще использовать уже готовые, отлаженные процедуры, включенные в состав операционной системы. В то же время даже при наличии в ОС соответствующей функции программист может реализовать ее самостоятельно в рамках приложения, если предложенный операционной системой вариант его не вполне устраивает. Возможности операционной системы доступны прикладному программисту в виде набора функций, называющегося интерфейсом прикладного программирования (Application Programming Interface, API). От конечного пользователя эти функции скрыты за оболочкой алфавитно-цифрового или графического пользовательского интерфейса. Для разработчиков приложений все особенности конкретной операционной системы представлены особенностями ее API. Поэтому операционные системы с различной внутренней организацией, но с одинаковым набором функций API кажутся им одной и той же ОС, что упрощает стандартизацию операционных систем и обеспечивает переносимость приложений между внутренне различными ОС, соответствующими определенному стандарту на API. Например, следование общим стандартам API UNIX, одним из которых является стандарт Posix, позволяет говорить о некоторой обобщенной операционной системе UNIX, хотя многочисленные версии этой ОС от разных производителей иногда существенно отличаются внутренней организацией. Приложения выполняют обращения к функциям API с помощью системных вызовов. Способ, которым приложение получает услуги операционной системы, очень похож на вызов подпрограмм. Информация, нужная ОС и

состоящая обычно из идентификатора команды и данных, помещается в определенное место памяти, в регистры и/или стек. Затем управление передается операционной системе, которая выполняет требуемую функцию и возвращает результаты через память, регистры или стеки. Если операция проведена неуспешно, то результат включает индикацию ошибки. Способ реализации системных вызовов зависит от структурной организации ОС, которая, в свою очередь, тесно связана с особенностями аппаратной платформы. Кроме того, он зависит от языка программирования. При использовании ассемблера программист устанавливает значения регистров и/или областей памяти, а затем выполняет специальную инструкцию вызова сервиса или программного прерывания для обращения к некоторой функции ОС. При использовании языков высокого уровня функции ОС вызываются тем же способом, что и написанные пользователем подпрограммы, требуя задания определенных аргументов в определенном порядке.