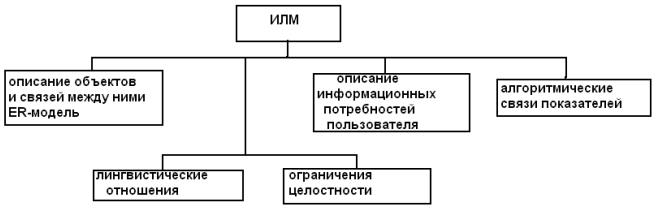
Билет 16

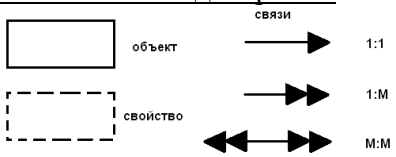
1) ER-модель данных. Основные нотации изображения ER-модели.

ER - модель – центральная компонента инфологической модели (позволяющей адекватно отображать предметную область). ER-модель описывает объекты и связи между ними.



Для описания ER-модели (объект – свойство - отношение) используют как языковые, так и графические средства. Объекты, имеющие одинаковый набор свойств, группируются в классы объектов со своими идентификаторами.

Элементы ER-диаграмм:



Свойства, не изменяющиеся во времени – статические (S), изменяющиеся – динамические (D).

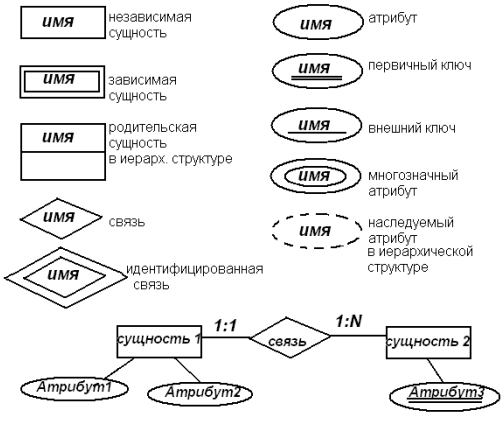
Класс принадлежности показывает, может ли отсутствовать связь объекта данного класса или она обязательна. В последнем случае добавляется разделитель с точкой. Например, изделие имеет в своем составе детали. Каждое изделие должно иметь хотя бы одну деталь, но не более чем одну.

Объекты могут быть простыми (неделимыми на составляющие) и сложными (составными (соответствуют отображению «целое-часть», например «группа-студенты»), обобщенными (связь «род-вид», например аспирант, школьник и студент образуют обобщенный объект «учащиеся»), агрегированными (соответствуют какому-либо процессу, куда вовлечены другие объекты, например, поставка деталей заводом заказчику, обозначается ромбом)).

Например:

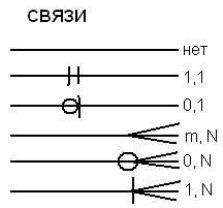
Существует несколько нотаций ER-модели.

Нотация Чена. Здесь прямоугольник, где написано имя – независимая сущность. Если прямоугольник двойной, то это зависимая сущность.



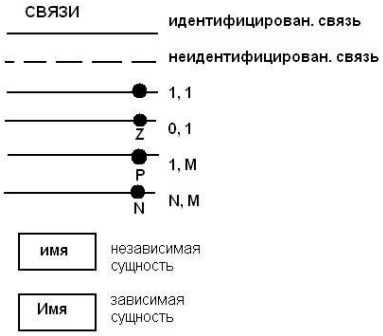
Нотация Мартина.

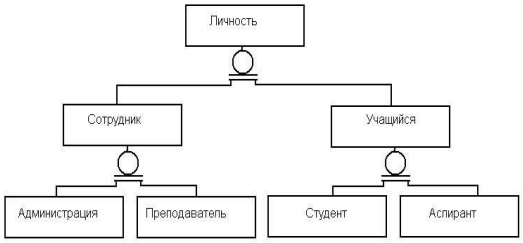
Независимая и родительские сущности аналогичны нотациям Чена.



Нотация IDEF1X:

Ключевые атрибуты прописываются в верхней части сущности.





Пример в нотации case oracle:

2) Геометрическое моделирование. Способы задания 3D объектов.

При формировании 3D модели используются:

- двумерные элементы (точки, прямые, отрезки прямых, окружности и их дуги, различные плоские кривые и контуры),
- поверхности (плоскости, поверхности, представленные семейством образующих, поверхности вращения, криволинейные поверхности),
- объемные элементы (параллелепипеды, призмы, пирамиды, конусы, произвольные многогранники и т.п.).

Два основных способа формирования геометрических элементов моделей - построение по заданным отношениям (ограничениям) и построение с использованием преобразований.

Построение с использованием отношений

Построение с использованием отношений заключается в том, что задаются:

- элемент подлежащий построению,
- список отношений и элементы к которым относятся отношения.

Используется два способа реализации построения по отношениям - общий и частный.

При общем способе реализации построение по заданным отношениям можно представить в виде двухшаговой процедуры:

- на основе заданных типов отношений, элементов и параметров строится система алгебраических уравнений,
- решается построенная система уравнений.

Достоинство:

- простота расширения системы
- для введения нового отношения достаточно просто написать соответствующие уравнения.

Недостатки:

- построенная система уравнений может иметь несколько решений, поэтому требуется выбрать одно из них, например, в диалоговом режиме,
- система уравнений может оказаться нелинейной, решаемой приближенными методами, что может потребовать диалога для выбора метода(ов) приближенного решения.

Частный подход - для каждой триады, включающей строящийся элемент, тип отношения и иные элементы, затрагиваемые отношением, пишется отдельная подпрограмма (например построение прямой, касательной к окружности в заданной точке). Требуемое построение осуществляется выбором из меню и тем или иным вводом требуемых данных.

Построение с использованием преобразований:

- задается преобразуемый объект,
- задается преобразование (это может быть обычное аффинное преобразование, определяемое матрицей, или некоторое деформирующее преобразование, например, замена одного отрезка контура ломаной),
- выполнение преобразования; в случае аффинного преобразования для векторов всех характерных точек преобразуемого объекта выполняется умножение на матрицу; для углов вначале переходят к точкам и затем выполняют преобразование.

3) Составить алгоритм поиска экстремума функции двух переменных $F(x_1, x_2) = x_1^4 + x_1^2 + x_1x_2 - 2x_2^2$ методом «тяжелого шарика»

Билет 17.

1) Представление знаний в экспертной системе

Методы представления знаний в ЭС: правила; семантические сети; фреймы. Возможно представление знаний в виде нечетких правил, а также в виде нейронных сетей.

Представление знаний в виде правил

Правила выражены условным отношением ЕСЛИ-ТО, т.е. когда условная часть правила удовлетворяет фактам, выполняется следственная часть.

Действия, предусмотренные следственной частью, могут состоять:

- в модификации фактов;
- во взаимодействии с внешней средой.

Механизм логического вывода в этом случае работает в цикле: сопоставить и выполнить.

Достоинства:

естественность, единообразие структуры, гибкость изменения логического вывода.

Недостатки:

процесс трудно поддаётся управлению, трудно представить иерархию понятий, малоэффективен процесс поиска решений.

Представление знаний с использованием фреймов

Фреймом называется структура для описания стереотипной ситуации, которая включает характеристики этих ситуаций и их значения.

Характеристики называются слотами, а их значения - заполнителями слотов. Заполнители слотов могут содержать не только значения, но и процедуру определения этого значения, в том числе и правило для поиска этого значения.

Процедуры, которые входят в состав слотов, бывают 3-х видов:

- *"Если добавлено"* . Процедура выполняется, если добавлена новая информация в слот.
- *"Если удалено"* . Процедура выполняется, если удаляется информация из слота.
- *"Если необходимо"*. Процедура выполняется, если запрашивается информация из слота, а он - пустой.

Процедуры, расположенные в слоте, называются связанными.

Фрейм представляет собой иерархическую структуру для моделирования конкретной предметной области. На верхнем уровне иерархии находится фрейм, содержащий информацию, истинную для всех остальных фреймов. Фреймы обладают способностью наследовать характеристики фреймов, находящихся на более высоком уровне, но они могут быть изменены на более низком уровне. Это позволяет обрабатывать изменения в знаниях.

Представление знаний в виде семантических сетей.

При разработке экспертных систем для графического представления знаний используются семантические сети. Метод основан на сетевой структуре. Семантические сети состоят из узлов, изображаемых окружностями, и соединяющих их линий со стрелками. Узлы сети обозначают некоторые порции информации, а соединяющие их линии - взаимосвязи между ними.

К данному виду представления знаний относятся следующие отношения:

- *"Является"* Объект входит в состав данного класса.
- *"Имеет"* Позволяет задавать свойства объектов.
- *"Является следствием"*.
- *"Имеет значение"*.

Достоинства:

лёгкость обработки иерархии отношений.

Недостаток:

сложность обработки исключений.

Представление знаний в виде нечетких высказываний

Методы построения математических моделей часто основаны хотя и не неточной, но в целом объективной информации об объекте. Однако возможны ситуации, когда при построении моделей решающее значение имеют сведения, полученные от эксперта, обычно качественного характера. Они отражают содержательные особенности изучаемого объекта и формулируются на естественном языке. Описание объекта в таком случае носит нечеткий характер.

Нейронные сети

Моделирование сложных систем требует большого числа знаний об объекте, в том числе экспериментальных и экспертных. Для их обработки в последнее время широко используются нейронные сети.

В основе теории нейронных сетей лежит желание воспроизвести функции мозга при решении конкретной задачи. Однако создающиеся системы не полностью воспроизводят функции мозга, а, скорее, представляют математическую модель, воспроизводящую отдельные возможности человеческого мозга ,по аналогии с которым искусственные нейтронные сети характеризуются следующими свойствами:

– обучение (т.е. изменение поведения в зависимости от окружающей среды).

– обобщение (реакция сети после обучения будет, до известной степени, нечувствительна к малым изменениям входящих сигналов).

– абстрагирование (способность выявления различий во входных сигналах).

Искусственный нейрон

Отдельный обрабатываемый элемент искусственной нейронной сети называется искусственным нейроном. Каждый нейрон производит относительно простую работу. На его вход поступает набор сигналов $X = [x1, x2, ..., xn]$, каждый из которых может быть выходом от другого нейрона или другого источника.

Каждый вход умножается на соответствующий угловой коэффициент $W = [w1, w2, ..., wn]$, который соответствует силе синапса биологического нейрона и поступает на вход суммирующего блока, где все произведения $w_i x_i$ суммируются. По этой величине определяется общий вход нейрона $h=\Sigma (w_i*x_i+Q_i)$, где θi – пороговая величина i-го нейрона.

Для определения выхода нейрона O (рис. 8) используется функция активации

$\theta = F(h)=F(\Sigma (w_i*x_i+Q_i))$ Наиболее типичными функциями активации являются:

– экспоненциальная

$$F(h)=\frac{2}{1+\exp(-\lambda h)}-1, \quad \lambda>0;$$

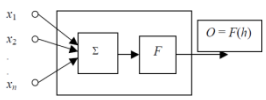


Рис. 8

– функция знака

$$F(h)=\operatorname{sgn}(h)=\begin{cases}+1, & h>0;\\ -1, & h<0.\end{cases} \qquad (6)$$

Функции, записанные в таком виде, называются биполярными. Возможно использование униполярных функций:

$$F(h)=\frac{1}{1+\exp(-\lambda h)}, \quad \lambda>0; \qquad (7)$$

$$F(h)=\begin{cases}1, & h>0;\\ 0, & h<0.\end{cases} \qquad (8)$$

Следует отметить, что при $\lambda\to\infty$ экспоненциальная функция приближается к функции знака.

2) Устройства автоматизированного считывания информации (сканеры). Конструкция, основные характеристики.

Сканером называется устройство, позволяющее вводить в компьютер в графическом виде текст, рисунки, слайды, фотографии и др.

Сканеры бывают настольные - они обрабатывают весь лист целиком, причем лист кладется внутрь сканера, либо вставляется в специальный механизм подачи, “проходит” через сканер и выходит с другой стороны. И ручные - их надо проводить над нужным рисунком или текстом. В зависимости от типа –

могут выдавать черно-белые или цветные изображения. Сканеры отличаются друг от друга разрешающей способностью, количеством воспринимаемых цветов или оттенков серого цвета.
Технология считывания данных в устройствах оцифровывания изображений реализуется на основе использования светочувствительных датчиков. Эти датчики преобразовывают интенсивность падающего на них отраженного света в пропорциональный ей электрический заряд. Также используется принцип усиления, отраженного от оригинала, ксенонового или вольфрамо-галогенного света. Этот свет, попадая на катод, выбивает из него электроны, которые вызывают вторичную электронную эмиссию на пластинах динодов. Напряжение, пропорциональное освещенности катода, снимается с анода и преобразуется в цифровой код.
Характеристики сканеров:
- Оптическое разрешение (Optical resolution) сканера измеряется в пикселах на дюйм (ppi - pixels per inch). Следует помнить, что часто используемый для описания оптического разрешения сканера термин dpi с технической точки зрения характеризует выходное разрешение сканированного изображения в зависимости от выбранного режима печати.
- Область сканирования (Scanning area) определяет размер самого большого оригинала, который может быть сканирован устройством
- Разрядность битового представления (Bit length representation) в качестве показателя степени 2 определяет максимальное число цветов или градаций серого, которые может воспринимать сканер. Для определения данного параметра цветных сканеров также используется термин глубина цвета (Color depth)
- Скорость сканирования (Scanning speed) — показатель быстродействия сканера, означает время, затрачиваемое на обработку одной строки оригинального изображения. Измеряется в миллисекундах (мс). На практике под скоростью сканирования понимают количество страниц черно-белого оригинала, сканируемых в минуту с максимальным оптическим разрешением
- Интерфейс (Interface) в описании сканера следует понимать варианты аппаратного подключения устройства к компьютеру

Для связи с PC сканеры могут использовать специальную 8- или 16-разрядную интерфейсную плату, вставляемую в соответствующий слот расширения.
В настоящее время широко используются стандартные интерфейсы, применяемые в IBM PC-совместимых компьютерах (последовательный и параллельный порты, а также интерфейс SCSI). В случае использования стандартного интерфейса, как правило, проблем с распределением системных ресурсов не возникает.

3) Составить программу для вычисления скорости передачи информации между компьютерами, объединенными в локальную сеть

```
//ПРОГРАММА СЕРВЕР
unit NetTestSrv;

interface

type
  TForm1 = class(TForm)
    Socket1: TServerSocket;
    procedure Socket1Read(Sender: TObject; Socket: TCustomWinSocket);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

type

implementation

procedure TForm1._FORM_CREATE(Sender: TObject);
begin
     Socket1.Port:=1203038;
```

```
    Socket1.Active:=True;
end;

procedure TForm1.Socket1Read(Sender: TObject; Socket: TCustomWinSocket);
begin
    Socket1.Socket.SendText(Socket.ReceiveText;);
end;

end.
//ПРОГРАММА КЛИЕНТ
unit NetTestClient;

interface
type
  TForm1 = class(TForm)
    Socket1: TClientSocket;
    procedure Socket1Read(Sender: TObject; Socket: TCustomWinSocket);
  end;

implementation

var i:integer;

procedure TForm1._FORM_CREATE(Sender: TObject);
begin
    Socket1.Address:='127.0.0.0';
    Socket1.Port:=530262;
    Socket1.Active:=True;
    i=GetTickCount();
    Socket1.Socket.SendText('TEST TEXT');
end;

end;

procedure TForm1.Socket1Read(Sender: TObject; Socket: TCustomWinSocket);
begin
   if Socket1.Socket.ReceiveText='TEST TEXT' then begin
       ShowMessage('Время передачи данных - ' + IntToStr(GetTickCount()-i) + ' мс');
   end;
end;
```

Билет 18.

1) Системно-сетевая телеобработка

Телеобработка данных — это такая организация информационно-вычислительного процесса, при которой ресурсы одной или нескольких ЭВМ одновременно используются многими пользователями через различные виды средств связи.

Системы телеобработки обеспечивают организацию двух основных методов обработки данных — пакетного и диалогового. Соответственно выделяются и два основных режима взаимодействия удаленного абонента с ЭВМ — режим пакетной передачи данных и диалоговый режим взаимодействия.

Системная телеобработка данных

Системная телеобработка определяется как взаимоувязанная совокупность технических и программных средств, обеспечивающая коллективное использование ресурсов и баз данных одной ЭВМ (вычислительного комплекса) большим количеством пользователей, подключенных к ЭВМ через средства связи и передачи данных. Основными средствами системной телеобработки данных являются телекоммуникационные методы доступа, мультиплексоры передачи данных, абонентские пункты, аппаратура передачи данных.

Сетевая телеобработка данных

Сетевая телеобработка определяется как взаимоувязанная совокупность унифицированных логических и физических средств, протоколов, интерфейсов, обеспечивающая возможность распределения управляющих и обрабатывающих мощностей, а также баз данных по сети. Такая телеобработка обеспечивает коллективное использование ресурсов и баз данных одной или нескольких территориально рассредоточенных ЭВМ большим количеством пользователей, подключенных к ЭВМ через средства связи и передачи данных.

Для описания взаимодействия компонентов в сети используются *протоколы и интерфейсы.*

Наиболее важными функциями протоколов на всех уровнях сетевой телеобработки являются защита от ошибок, управление потоками данных в сети, защита сети от перегрузки и выполнение операций по маршрутизации сообщений и оптимизации использования ресурсов в сети.

2) Тестирование программ.

Тестирование – предназначено для выявления грубых ошибок, которые не нарушают работоспособность программы, но не дают ей возможности выдавать правильный результат.

Для выявления подобного несоответствия необходимо заготовить тесты, которые не только выявляют ошибочность функционирования, но и позволяют локализовать подозрительное на ошибку место программы.

Дейкстра, в 1968 году заявил о необходимости исключить из состава языков программирования высокого уровня оператора goto. Позднее была сформулирована четкая система правил, названная структурным программированием.

Эта система представляет собой ряд ограничений и правил, которые обеспечивают соответствие любой программы строгому образцу, исключая запутанность, порождающую ошибки и затрудняющую тестирование. Одним из главных вопросов, побудивших Дейкстру к новой методике стала необходимость облегчения тестирования программных комплексов в наиболее полном объеме. Необходимость в этой процедуре вызвана следующим:

1) трудоемкость и стоимость тестирования больших программных комплексов возрастает экспоненциально с увеличением их размеров. При этом есть мнение, что стоимость проверки любого изменения в сложном комплексе более чем в сто раз превышает стоимость внесения этого изменения.

2) В сложных системах, требующих постоянного сопровождения и развития, ошибки присутствуют всегда

3) Издержки от испытания сложных комплексов постоянно возрастают, по мере того, как люди возлагают на ЭВМ все более ответственные функции.

Одним из выходов может стать разработка сложных и дорогостоящих методов тестирования программ (их эффективность является спорной). Дейкстра заложил в своей методике автоматическое доказательство правильности создаваемой программы, то есть требование правильности программы выполняется на любом этапе ее конструирования. Структурное программирование обладает тем дополнительным преимуществом, что повышает читаемость программы.

3) Исследовать на экстремум функцию: $F(x)= x_1^2 +3x_2^2 + 2x_3^2$, при условии $x_1 - x_2 + x_3 = 4$

Составляем функцию Лагранжа

$$L(\lambda, x) = f_0(x) + \sum_{j=1}^{m} \lambda_j f_j(x)$$

Достаточное условие:

- равенству нулю частных производных ф-ии Лагранжа по всем переменным.
- матрица вторых производных положительно определена.

$$L = x_1^2 + x3_2^2 + 2x_3^2 + \lambda(x_1 + x_2 + x_3 - 4)$$

$$\frac{\partial L}{\partial x_1} = 0; \frac{\partial L}{\partial x_1} = 2x_1 + \lambda = 0$$

$$\frac{\partial L}{\partial x_2} = 0; \frac{\partial L}{\partial x_2} = 6x_2 + \lambda = 0$$

$$\frac{\partial L}{\partial x_3} = 0; \frac{\partial L}{\partial x_3} = 4x_3 + \lambda = 0$$

$$\frac{\partial L}{\partial \lambda} = x_1 + x_2 + x_6 - 4 = 0$$

Решая систему, находим ***$x_1 = 2,182; x_2 = -0,727; x_3 = 1,091$*** .

Матрица вторых производных :

$$H = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 4 \end{bmatrix} = 48$$, т.е. определена отрицательно, это точка min.

Билет 19.

1) Операционная среда ЭВМ и САПР. Системное программирование. Типы операционных систем.

Операционная система - набор сервисных программ, выполняющих две основные функции:

1) управление ресурсами системы, их распределением между несколькими пользователями и контроль за выделением ресурсов для одновременного выполнения нескольких задач;

2) предоставление набора услуг обеспечивающие пользователю наибольшие удобства в общении с ЭВМ.

Способы классификации ОС:

- по количеству пользователей одновременно обслуживаемых системой - однопрограммная и мультипрограммная (многопользовательская) система.

Основная цель мультипрограммирования - это увеличение производительности системы за счет разделения ее ресурсов между несколькими заданиями.

ОС предоставляет пользователю набор команд, посредством которых он может получать доступ к системным ресурсам. Дальнейшее развитие выч. техники повлекло за собой изменение интерфейса пользователя. Общение с ОС посредством системы стало крайне неудобным. Поэтому на смену такому интерфейсу приходит интерфейс с использованием графических образов. Пользователь оперирует с графич. образами каких-либо объектов понятных ему. При этом ОС все манипуляции пользователя интерпретирует в понятном для себя виде (концепция ОС фирмы Apple).

Важная функция ОС - поддержка операционного окружения (ОО) пользовательских задач. Оно состоит из ряда стандартных сервисных программ, которые могут быть использованы в процессе выполнения задачи и предоставлять средства для управления ресурсами выч. системы, выделяя их пользователю по мере надобности.

2) Автоматизация функционально-логического этапа проектирования цифровых узлов и устройств.

На современном уровне развития технологии проектирования достигнута значительная степень автоматизации функционально-логического этапа проектирования цифровых узлов и устройств. Если раньше при проектировании главным инструментом инженера был паяльник, то теперь ему на смену пришли современные пакеты прикладных программ. Одной из таких помощниц является программа MicroCap, которую мы изучали на схемотехнике. Вместо использования реальных электрических схем (а в этом случае велики затраты и вероятность выхода схемы из строя из-за ошибок инженера) используются их математические модели. Это упрощает, ускоряет и облагораживает процесс проектирования.

3) Графические форматы. BMP, GIF и JPEG.

BMP — официальный графический формат платформы Windows.

Он хранит данные о цвете только в модели rgb, поддерживает, как индексированные цвета, так и true color, причем в режиме индексированных цветов возможна простейшая компрессия RLE.

Вся «мультиплатформенность» формата заключается лишь в поддержке Windows и OS/2.

Чтобы восстановить графический образ на экране из формата bmp не надо проводить никаких сложных и ресурсоемких операций по декодированию — достаточно лишь последовательно считывать номера цветов пикселей в палитре rgb и отображать их поток на экране. Такой простой алгоритм не может не сказаться на степени загрузки процессора при обработке файлов bmp.

Используют: для хранения логотипов, splash-screen'ов, иконок и прочих графических бирюлек внутри программ.

GIF - популярный формат графических изображений.

Область применения

Изображение в формате GIF хранится построчно, поддерживается только формат с индексированной палитрой цветов. Стандарт разрабатывался только для поддержки 256-цветовой палитры.

Один из цветов в палитре может быть объявлен «прозрачным».

Анимированные изображения

Формат GIF поддерживает анимационные изображения. Они представляют собой последовательность из нескольких статичных кадров, а также информацию о том, сколько времени каждый кадр должен быть показан на экране

Сжатие

GIF использует формат сжатия LZW. Таким образом, хорошо сжимаются изображения, строки которых имеют повторяющиеся участки. В особенности изображения, в которых много пикселей одного цвета по горизонтали.

Чересстрочный GIF

Формат GIF допускает чересстрочное хранение данных. При этом строки разбиваются на группы, и меняется порядок хранения строк в файле. При загрузке изображение проявляется постепенно, в несколько проходов. Благодаря этому, имея только часть файла, можно увидеть изображение целиком, но с меньшим разрешением.

Альтернатива

Существует формат APNG, созданный в 2004 году, использующий 24-битные цвета и 8-битную полупрозрачность, работающий в браузерах Mozilla Firefox и Opera начиная с 2007 года. Некоторые программы и расширения также поддерживают APNG.

JPG

К 1991 году мировая интернет-общественность осознала, что негоже пытаться подменить все буйство красок окружающего мира жалкими индексированными цветами gif. Проблему изучили капитально. В результате получился достаточно сложный по сравнению с другими алгоритм компрессии jpg и одноименный формат вместе с ним.

Используется: представление блока пикселей 8x8 одним цветом с сохранением информации о яркости, плюс метод Хаффмана и многое другое в зависимости от степени компрессии.

JPG отлично сжимает фотографии, но это сжатие происходит с потерями качества. Впрочем, чаще всего это ухудшение совершенно незаметно, кроме случаев, когда вокруг контуров с резкими переходами цвета образуются своеобразные помехи.

Существует три подформата jpg: обычный, optimized (файлы несколько меньше, но не поддерживаются старыми программами) и Progressive (чересстрочное отображение, аналог interlaced в gif).

Некоторые приложения позволяют хранить изображение в jpg в режиме CMYK и даже включать в файл обтравочные контуры.

Однако, использовать jpg для полиграфических нужд категорически не рекомендуется из-за взаимодействия регулярной структуры блоков 8x8 пикселей, получающихся в результате компрессии, с не менее регулярной структурой типографского растра, что в итоге приводит к образованию муара.

Из долговременного пользования этим безусловно полезным форматом можно извлечь две вещи:

- не стоит сохранять в нем все, что попало, а только крупные фотографии с большим количеством плавных цветовых переходов.
- ни в коем случае не стоит сохранять одно и то же изображение в jpg больше одного раза: слишком заметными оказываются деструктивные изменения картинки от повторного использования компрессии

Билет №20

1) Понятие алгоритма. Свойства. Способы записи.

Алгоритмом называется точное предписание, определяющее точное содержание и порядок действий, которые необходимо выполнить над исходными данными для получения конечного результата при решении всех задач определенного класса.

Свойства алгоритма:

1. Массовость (для решения целого количества задач)
2. Понятность
3. Дискретность
4. Конечность
5. Определенность (при исполнении алгоритма исполнитель должен соответствовать с заданными действиями)
6. Эффективность.

Способы записи алгоритма.

1) Словесная запись.

Это перечисление словесных действий, которые необходимо выполнить в определенном порядке.

Пример:

$$y = \begin{cases} -2, & x < 1 \\ -x + 3, & x \geq 1 \end{cases}$$

1. Ввести значение x
2. Если x<1, то y=-2, иначе y=-x+3
3. Печать y
4. Конец

2) Решающая таблица

Состоит из 4-х частей: перечень условий, перечень действий, указатель условий, указатель действий.

Таблица	1	2	...	K	
Условие 1	Y	N	...	Y	указатели условий
...					
Условие N	N	Y	...	N	
Действие 1	X	...	...	...	указатели действий
...					
Действие 2	...	...	...	X	

Содержание столбца составляет правила алгоритма, определяющие какие условия следует проверить, каким должен быть результат проверки и какие действия.

Если условие входит как элемент из правила - ставится Y(yes - условие должно быть выполнено) или N(no - не должно).

В том случае, когда правила требуют выполнения некоторых действий на пересечении строки и столбца ставится X.

Пример:

Таблица	1	2	
x<1	Y	N	Если выполняется условие x<1, то
y=-2	X		y=-2 и печатаем y (столбец 1).

y=x+3		X	Если не выполняется условие x<1, то
печать y	X	X	y=x+3 и печатаем y (столбец 2).

2) Построение реалистичных изображений. Алгоритм построения теней в машинной графике.

Самый простой способ закрашивания называется гранением или однотонной закраской. Он требует сравнительно небольших ресурсов компьютера, поскольку предполагает лишь заполнение каждого из многоугольников одним цветом или оттенком, который определяется с использование лишь одной нормали - нормали к рассматриваемому многоугольнику. Однако такой способ закраски слишком примитивен; закрашенные этим способом объекты выглядят не плавными, а так как если бы они были покрыты рыбными чешуйками.

Алгоритм закраски по методу Гуро

Более реалистические изображения получаются в случае закрашивания методом Гуро. При таком способе закраски яркость и цветовая насыщенность каждого многоугольника плавно меняется не только от угла к углу, но и вдоль его ребер. Такое закрашивание осуществляется в четыре этапа:

* вычисление нормалей к граням;
* определение нормалей в вершинах путём усреднения нормалей по граням, которым принадлежит данная вершина;
* вычисление интенсивности вершин;
* закрашивание многоугольника, при котором интенсивность каждого его пиксела определяется путем билинейной интерполяции значений интенсивности в вершинах.

Основной недостаток - эффект полосы Маха: на ребрах смежных многоугольников возникает полоса разрыва непрерывности. Это происходит потому, что такой метод интерполяции обеспечивает лишь непрерывность значений интенсивности вдоль границ многоугольников, но не обеспечивает непрерывности изменения интенсивности.

Алгоритм закраски по методу Фонга

Закраска Фонга требует больших вычислительных затрат, однако она позволяет разрешить многие проблемы метода Гуро, в том числе и проблему полосы Маха, поскольку предполагает плавное изменение яркости и насыщенности не только вдоль ребер каждого многоугольника, но и по самой поверхности. При закраске Гуро вдоль сканирующей строки интерполируется значение интенсивности, а при закраске Фонга - вектор нормали. Затем она используется в модели освещения для вычисления интенсивности пиксела. При этом достигается лучшая локальная аппроксимация кривизны поверхности и, следовательно, получается более реалистичное изображение. В частности, правдоподобнее выглядят зеркальные блики.

Хотя, метод Фонга устраняет большинство недостатков метода Гуро, он тоже основывается на линейной интерполяции. Поэтому в местах разрыва первой производной интенсивности заметен эффект полос Маха, хотя и не такой сильный, как при закраске Гуро. Однако, иногда этот эффект проявляется сильнее у метода Фонга, например для сфер. Кроме того, оба метода могут привести к ошибкам при изображении невыпуклых многоугольников.

Также возникают трудности, когда любой из этих методов применяется при создании последовательности кинокадров. Например, закраска может значительно изменяться от кадра к кадру. Это происходит из-за того, что правило закраски зависит от поворотов, а обработка ведётся в пространстве изображения. Поэтому, когда от кадра к кадру меняется ориентация объекта, его закраска (цвет) тоже изменяется, причём достаточно заметно.

3) Записать алгоритм определения объема кэш-памяти первого уровня.

Описание

1.Создаем квадратную матрицу размера n на n, где n вводится пользователем. Присваиваем элементам матрицы случайные числа. Присвоим start1 время работы вызывающей программы. Затем складываем элементы матрицы по строкам и end1 присвоим время после сложения. Преобразуем разницу времен между end1и start1 в секунды. Присвоим start2 время работы вызывающей программы. Складываем элементы матрицы по столбцам и end2 присвоим время после сложения.

Преобразуем разницу времен между end2и start2 в секунды. Видим что time1 и time2 различны, значит произошло переполнение КЭШ памяти, которое вызвало ее перезагрузку.
2. Создаем массив размером n^2 . Присваиваем элементам массива случайные числа. Присвоим start3 время работы вызывающей программы. Затем складываем элементы массива последовательно и end1 присвоим время после сложения. Преобразуем разницу времен между end1и start1 в секунды. Присвоим start4 время работы вызывающей программы. Складываем элементы массива хаотично и end2 присвоим время после сложения. Преобразуем разницу времен между end2и start2 в секунды. Видим что time3 и time4 различны, значит произошло переполнение КЭШ памяти, которое вызвало ее перезагрузку.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

 void main (void) {
      const  unsigned  long long int step=80000;
      clock_t start1, end1, start2, end2;
      unsigned  long long int i,j;
      int n;
      double time1,time2;
      printf("Enter size of matrix: ");
      scanf("%d", &n);
      unsigned  long long int **mat = (unsigned  long long int**)calloc(n,sizeof(unsigned  long long int*));
      for(i=0; i<n; i++)
      mat[i] = (unsigned  long long int*)calloc(n,sizeof(unsigned  long long int));
      unsigned  long long int sum1=0, sum2=0;
      srand(time(NULL));
      for(i=0; i<n; i++)
               for(j=0; j<n; j++) {
                        mat[i][j]= rand();
                }
      start1=clock();
      unsigned  long long int k;

      for(k=0;k<step;k++){
               for(i=0;i<n;i++){
                        for(j=0;j<n;j++){
                                 sum1=(sum1+mat[i][j]);
                        }
               }
      }
      end1=clock();
      time1=((end1-start1)/CLOCKS_PER_SEC);
      printf("Time of first adding all the elements of the matrix by rows: %f\n", time1);
      start2=clock();
      for(k=0;k<step;k++){
               for(j=0;j<n;j++){
                        for(i=0;i<n;i++){
                                 sum2=(sum2+mat[i][j]);

                        }
               }
      }
      end2=clock();
      time2=((end2-start2)/CLOCKS_PER_SEC);
      printf("The second addition of all the elements of the matrix by columns: %f\n", time2);
      printf("----------------------------\n");
      unsigned  long long int *mat2 = (unsigned  long long int*)calloc(n*n, sizeof(unsigned  long long int));
      unsigned  long long int *mat3 = (unsigned  long long int*)calloc(n*n, sizeof(unsigned  long long int));
```

```
    for(i=0;i<n*n;i++)
            mat2[i]=rand()%(n*n);
    double time3,time4;
    clock_t start3, end3, start4, end4;
    unsigned  long long int sum3=0, sum4=0;
    start3=clock();

    for(k=0;k<step;k++){
            for(i=0;i<n*n;i++)
                    sum3=(sum3+mat2[i]);
            }
    end3=clock();
    time3=((end3-start3)/CLOCKS_PER_SEC);
    printf("Time of first adding all the elements of the array: %f\n", time3);
    start4=clock();
    for(k=0;k<step;k++){
            for(j=0;j <n*n;j++)
                    sum4=(sum4+mat3[mat2[j]]);
    }
    end4=clock();
    time4=((end4-start4)/CLOCKS_PER_SEC);
    printf("The second addition of all the elements of the array  randomly: %f\n", time4);

}
```

Билет №21

1) Проектирование программ: связность и цельность программных модулей.

Программное обеспечение проектируется по следующим принципам:

1. Принцип системного единства
2. Принцип развития
3. Принцип совместимости
4. Принцип стандартизации

Принцип системного единства подразумевает, что при создании, развитии и функционировании САПР связи между компонентами ПО должны обеспечивать ее целостность.
Принцип развития. ПО САПР должно создаваться и функционировать с учетом пополнения, совершенствования и обновления ее компонентов.
Принцип совместимости. Языки, символы, коды, информация и связи между компонентами системы должны обеспечивать их совместное функционирование и сохранять открытую структуру системы в целом.
Принцип стандартизации. При проектировании ПО САПР необходимо максимально унифицировать, типизировать и стандартизировать ПО, которое должно быть инвариантным (независимым) к проектируемым объектам.

2) Постоянные запоминающие устройства. Область применения.

ПЗУ предназначены для хранения постоянной или редко изменяющейся информации, которую можно считать также просто, как из ОЗУ.
По архитектурным принципам и функциональному назначению ПЗУ делятся на 2 основных группы: ПЗУ и ПЛМ (Программируемые логические матрицы).
Все полупроводниковые запоминающие устройства, в том числе и ПЗУ, представляют собой особую разновидность логических схем, общими признаком построения которых является регулярная матричная структура состоящая из матриц «И» и «ИЛИ». В ПЗУ информация записывается в матрицу «ИЛИ», матрица «И» представляет собой дешифратор всех 2^n выходов от входных комбинаций. В ПЛМ информация заносится либо в матрицу «И», либо в обе матрицы. Следует отметить, что ПЗУ и ПЛМ, у которых программируется одна матрица, относится только к одноуровневой матричной логике.
По способу записи информации ПЗУ делятся на однократно и многократно программируемые. К однократно программируемым относятся ПЗУ:
а) с масочным программированием (МПЗУ, МПЛМ)
б) программируемые потребителем ПЗУ (ППЗУ)
в) программируемые потребителем логические матрицы (ППЛМ)
г) программируемые потребителем матричная логика (ППМЛ)
Многократно программируемая или репрограммируемая матричная логика РПЗУ:
а) стирание ультрафиолетом (СППЗУ, СППЛМ)
б) электрически стираемая программируемая пользователем (ЭСППЗУ)

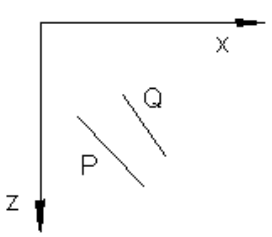
По способу считывания ПЗУ делятся на синхронные и асинхронные. По техническому изготовлению делятся на биполярные схемы и МОП, которые делятся по уровням входных и выходных сигналов.

3) Приоритетные методы удаления скрытых поверхностей. BSP – деревья.

Алгоритм, использующий список приоритетов.

Начало данного алгоритма – алгоритм художника.
Основная схема алгоритма:

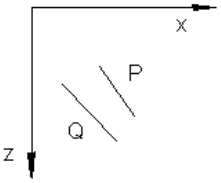
1. для каждого полигона сцены ищем zmin и zmax
2. сортируем все полигоны в порядке возрастания zmin
3. назовем самый первый левый полигон P, следующий – Q
4. если $z^P \max < z^Q \min$, то P и Q не перекрываются, выводим в растр P.

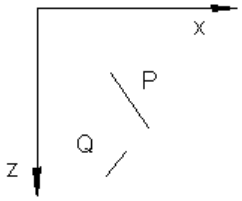


5. иначе $z^P \max > z^Q \min$, если такие полигоны есть, то они образуют список, в этом списке полигоны Q могут экранировать полигоны P. Для выявления этой ситуации предлагается 5 тестов, на которые необходимо ответить да или нет.

Если в любом тесте будет дан ответ да, P выводим в растр.

Тесты:

1. верно ли, что габариты P и Q не пересекаются по оси x
2. верно ли, что габариты P и Q не пересекаются по оси y
3. 

верно ли, что P целиком лежит по ту сторону плоскости, в которой лежит Q, которая находится дальше от камеры.
4. 

верно ли, что полигон Q лежит целиком до плоскости, в которой лежит P

для выполнения 3, 4 теста необходимо воспользоваться тестовой функцией плоскости.

Пусть плоскость задана уравнением вида

$Ax + By + Cz + D = 0$

$T = Ax + By + Cz + D$, подставляем координаты точки, если $T >= 0$, то точка за плоскостью, иначе перед ней.

5. верно ли, что проекции полигонов P и Q не пересекаются

Если ни один из тестов не дает ответ “да”, то меняют местами в списке P и Q. После этого выполняют 5 тестов. Иногда это помогает. Если в этом случае не помогло, то произошло зацикливание. В этом случае P разбивает Q на 2 части P1 и P2. После разбиения переходим к первому пункту алгоритма.

Для повышения эффективности данного алгоритма необходимо предварительно убирать нелицевые полигоны.

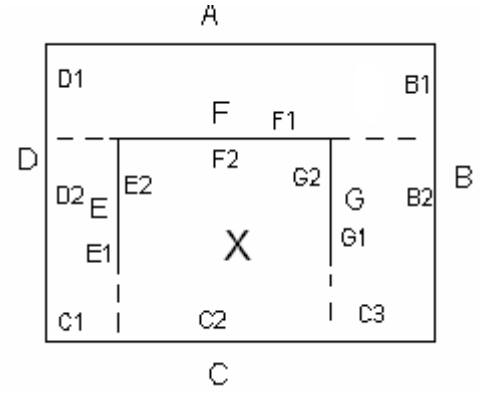
BSP-деревья

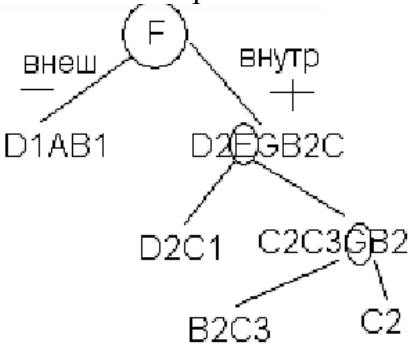
BSP – бинарное разбиение пространства

BSP дерево - это в сущности некая структура данных, которая будучи построена один раз для некоторого 3D обьекта позволяет потом без особых затрат времени сортировать по удаленности поверхности этого 3D обьекта при рассмотрении его с разных точек зрения.

Пример: построим BSP-дерево для данного помещения

Возьмем в качестве корневого элемента сторону F





Для показа BSP-дерева можно воспользоваться алгоритмом:

Например, мы находимся в точке X. Обращаемся в корневой элемент BSP-дерева. Если мы находимся внутри по отношению к F, то мы идем по дереву во внешнюю ветвь. Выписываем элементы ветви в отдельный список D1AB1. Далее поднимаемся в F и выписываем значение F. Идем в правую ветвь в узел E. Если мы находимся внутри по отношению к E, то идем во внешнюю ветвь. Выписываем D2C1EB2C3GC2.

Определение понятия внешней стороны:

Пусть требуется определить расположение полигона 2 и 3 по отношению к полигону 1. Для этого используют тестовую функцию:

$T = Ax + By + Cz + D$

Если в T полигона 1 подставить узловые точки полигона 2 и все значения будут положительны, то полигон 2 относится к правой ветви BSP-дерева, если отрицательны – то к левой. Если знаки меняются, то полигоны пересекаются.

Алгоритм визуализации BSP-дерева

Если камера находится в положительном подпространстве сцены по отношению к корневому элементу, то выводим отрицательную ветвь, далее корневой элемент и в конце выводим правую ветвь и это делается для каждого узла дерева рекурсивно.

Билет 22

1) Определение, состав и функции операционной системы.

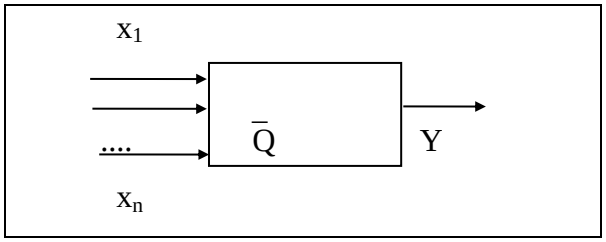
Операционная система - набор сервисных программ, выполняющих две основные функции:

- управление ресурсами системы, их распределением между несколькими пользователями и контроль за выделением ресурсов для одновременного выполнения нескольких задач;
- предоставление набора услуг обеспечивающие пользователю наибольшие удобства в общении с ЭВМ.

Способы классификации ОС:
по количеству пользователей одновременно обслуживаемых системой - однопрограммная и мультипрограммная (многопользовательская) система.
Основная цель мультипрограммирования - это увеличение производительности системы за счет разделения ее ресурсов между несколькими заданиями.
ОС предоставляет пользователю набор команд, посредством которых он может получать доступ к системным ресурсам. Дальнейшее развитие выч. техники повлекло за собой изменение интерфейса пользователя. Общение с ОС посредством системы стало крайне неудобным. Поэтому на смену такому интерфейсу приходит интерфейс с использованием графических образов. Пользователь оперирует с графич. образами каких-либо объектов понятных ему. При этом ОС все манипуляции пользователя интерпретирует в понятном для себя виде (концепция ОС фирмы Apple).
Важная функция ОС - поддержка операционного окружения (ОО) пользовательских задач. Оно состоит из ряда стандартных сервисных программ, которые могут быть использованы в процессе выполнения задачи и предоставлять средства для управления ресурсами выч. системы, выделяя их пользователю по мере надобности.

2) Методы проверки работоспособности жизненно важных объектов на этапе проектирования.

При работе спроектированного оборудования входные параметры могут отличаться от номинальных значений, использованных при проектировании. Кроме того при изготовлении деталей и узлов их параметры также будут иметь определенные погрешности. Влияние этих факторов может привести к неработоспособности объекта. Поэтому разработаны статистические методы использующие мат. модель для проверки работоспособности объекта на этапе проектирования.
1. Метод наихудшего случая



$\bar{Q}$ ={q_1, q_2,..., q_m}-вектор внутренних характеристик (размеры, теплоемкость, сопротивление, емкость, и т.д.).
Пусть известны максимальные отклонения вх. координат и внутренних переменных от номинальных значений:

$$x_{1\min} \le x_1 \le x_{1\max}$$
$$x_{2\min} \le x_2 \le x_{2\max} \qquad q_{1\min} \le q_1 \le q_{1\max}$$
$$\ldots\ldots \quad , \quad \ldots\ldots$$
$$x_{n\min} \le x_n \le x_{n\max} \qquad q_{m\min} \le q_m \le q_{m\max}$$

Пусть известно минимальной и максимальное выходной координаты y_{min} и y_{max} , при выполнении условия $y_{min} \le y \le y_{max}$ объект считается работоспособным.
Согласно методу наихудшего случая перебираются все сочетания крайних (наихудших) значений вх. координат и внутренних переменных. При каждом сочетании рассчитывается y и проверяется условие $y_{min} \le y \le y_{max}$.
Недостатки:
1) максимальные отклонения вх. координат и внутренних переменных от номинальных значений могут быть неизвестны.

2) вероятность одновременного сочетания всех наихудших значений в реальной жизни невелика.
Поэтому, по методу наихудшего случая может быть забракован принципиально работоспособный объект или могут быть приняты рекомендации увеличивающие габариты, массу, стоимость изделия.
Жизненно важные объекты проверяются на работоспособность именно по методу наихудшего случая.

2. Метод имитационного моделирования

Имитационно моделирование - проведение численных экспериментов на мат. модели с целью анализа функционирования проектируемого объекта.

Имитационная система включает генератор случайных процессов, математические модели и блок анализа результата (рис. 1.1).

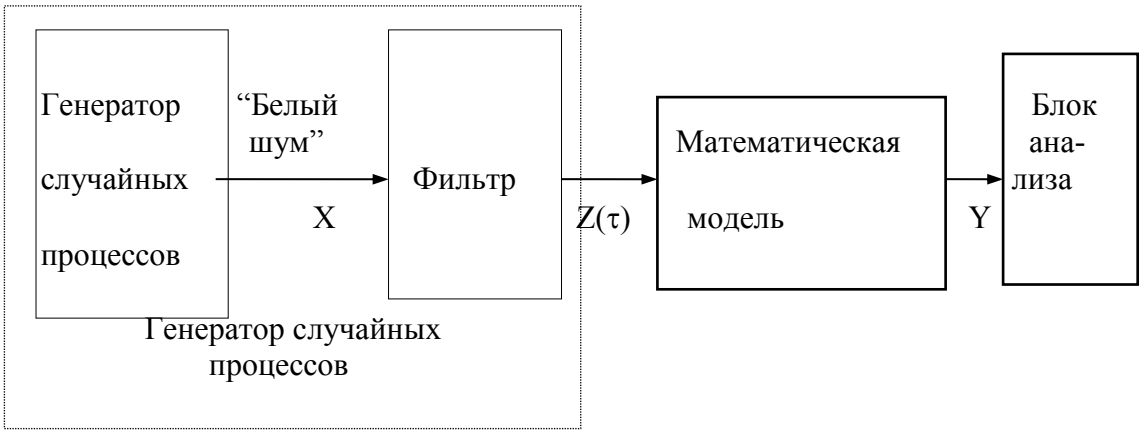


Рис.1 *Схема системы имитационного моделирования*

Генератор случайных процессов - программный модуль, на выходе которого имеется последовательность чисел, являющаяся случайным процессом с заданными мат. ожиданием M_0, дисперсией σ_0^2, и корреляционной функцией $K_0= \sigma_0^2 e^{-\alpha_0 S}$.

Построение генератора случайных процессов начинается с получения реальных значений случайного процесса на экспериментальной установке или на действующем промышленном объекте. Полученная реализация случайного процесса статически обрабатывается для нахождения мат. ожидания, дисперсии и корреляционной функции.

$$Z(\tau) \cong \frac{1}{N}\sum_{i=1}^{N} x(N-i)\sqrt{\frac{G^2}{G_x^{\,2}\alpha_0 A_2}}A_1 e^{-A_2\alpha_0 i} + M_0,$$

где A_1, A_2 - параметры, определяемые для каждого конкретного генератора случайных чисел.

Параметры A_1 и A_2 в формуле фильтра подбираются таким образом чтобы на выходе генератора случайных процессов получили заданные характеристики α_0, M_0, σ_0^2.

После того как последовательность чисел с ГСП подана на вход ММ, на выходе ММ получается последовательность чисел выходной координаты Y которая подается на блок анализа. Так же как и в методе наихудшего случая должны быть известны Y_{min}, Y_{max} при которых объект считается работоспособным.

Негативными признаются следующие результаты имитационного моделирования:

- выходная координата монотонно увеличивается(уменьшается) во времени и выходит за допустимые пределы Y_{max} (Y_{min}).
-выходная координата имеет отдельные выбросы, выходящие за пределы Y_{max},Y_{min}.
- скорость изменения выходной координаты превышает допустимое значение.

3) Записать алгоритм решения уравнения методом простых итераций: 3x-cos(x)+l =0. Проверить сходимость метода

Решение:

Приведем к виду $x = \varphi(x)$, тогда уравнение примет вид x=(cos(x)-1)/3;

$\varphi'(x) = -\sin(x)/3$

$|\varphi'(x)| < 1$, следовательно метод сходится, приведем алгоритм

1. $x_1 = x_0 = 0$; // стартовая точка, выбираемая произвольно
2. $x_0 = x_1$; // это для последующих возвратов к этому шагу
3. $x_1 = \varphi(x_0)$
4. если $|x_1 - x_0| > 0.0001$, то переходим к шагу 2
5. Вывод решения x_1 с точностью 0.0001

Билет 23

1) Функциональные узлы последовательного типа: регистры, триггеры, счетчики.

Последовательными логическими схемами называют полные цифровые автоматы, выходные сигналы которых зависят не только от состояний входных сигналов в текущий момент времени, но и от состояния схемы в предыдущий момент времени

Триггер – это элемент цифрового устройства с двумя устойчивыми состояниями. Под воздействием входного сигнала триггер может переключаться из одного положения в другое, при этом напряжение на его выходе скачкообразно изменяется.

Как правило, триггер имеет два выхода прямой и инверсный(Q, $\bar{Q}$). Число входов зависит от структуры и функций, выполняемых триггером. Входы, как и сигналы, подаваемые на них делятся на информационные и вспомогательные. Информационные сигналы через соответствующие входы управляют состоянием триггера. Сигналы на вспомогательных входах служат для предварительной установки триггера в заданное состояние и его синхронизации. Вспомогательные входы могут при необходимости выполнять роль информационного.

Входы и выходы триггера, как и соответствующие им сигналы, обозначают буквами:

S – раздельный вход установки в единичное состояние (напряжение высокого уровня на прямом входе Q);

R – раздельный вход установки в нулевое состояние (напряжение низкого уровня на прямом входе Q);

Д – информационный вход (на него передается информация, предназначенная для занесения в триггер);

С – вход синхронизации;

Т – счетный вход.

Триггеры классифицируют по ряду признаков. По функциональным возможностям выделяют триггеры: с раздельной установкой "0" и "1" (RS – триггеры); с приемом информации по одному входу (Д – триггеры); счетный Т – триггер; универсальный IK – триггер.

По способу приема информации триггеры подразделяют на асинхронные и синхронные. Асинхронные триггеры реагируют на информационные сигналы в момент их появления на входе. Синхронные – при наличии разрешающего сигнала специально предусмотренном входе С.

Счетчиком называют устройство, сигналы, на входе которого в определенном коде отображают число импульсов, поступивших на счетный вход. Триггер Т-типа может служить примером простейшего счетчика. Такой счетчик считает до двух. Счетчик, образованный цепочкой из m-триггеров, сможет посчитать в двоичном коде 2^m импульсов. Каждый из триггеров цепочки называют разрядом счетчика. Число m определяет количество разрядов двоичного числа, которое может быть записано в счетчик. Число $K_a=2^m$ называют коэффициентом (модулем) счета.

Информация снимается с прямых и (или) инверсных выходов всех триггеров. В паузах между входными импульсами триггеры сохраняют свое состояние, т. е. счетчик запоминает число сосчитанных импульсов.

Нулевое состояние всех триггеров принимается за нулевое состояние счетчика в целом.

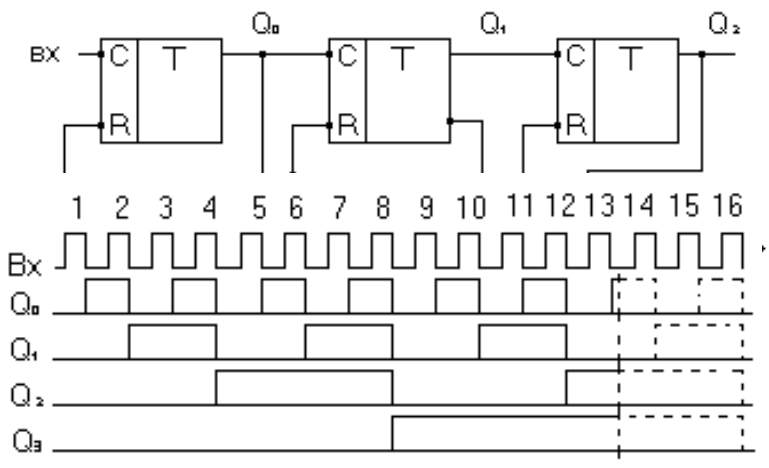
Классификация счетчиков.

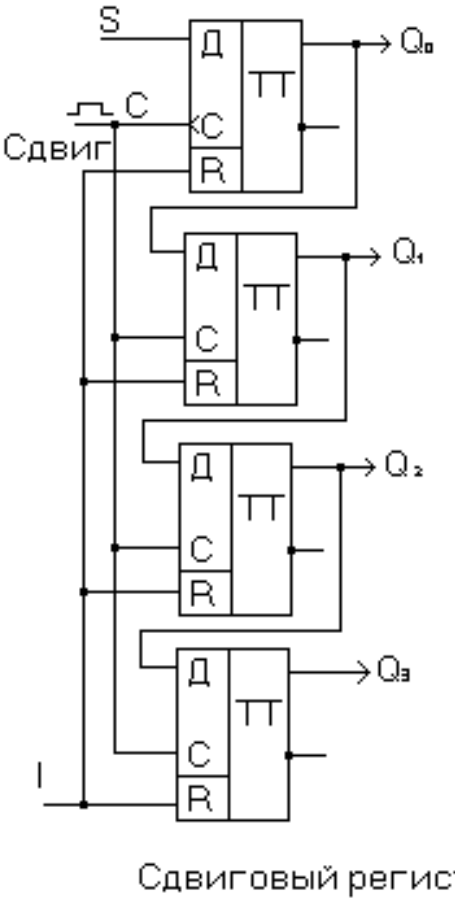
По коэффициенту счета: двоичные; двоично-десятичные (декадные) или с другим основанием счета; с произвольным постоянным модулем; с переменным модулем.

По направлению счета: суммарные; вычитающие; реверсные.

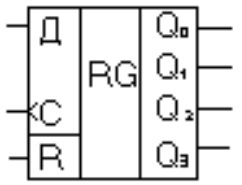
По способу организации внутренних связей: с последовательным переносом; с параллельным переносом; с комбинированным переносом; кольцевые.

Счетчик с параллельным переносом строят из синхронного триггера. Счетные импульсы подаются одновременно на все тактовые входы, а каждый из триггеров цепочки служит по отношению к последующим только источником информации. Срабатывание триггеров параллельного счетчика происходит синхронно и задержка переключения всего счетчика равна задержке переключения для одного триггера. Счетчики с параллельным переносом применяют в быстродействующих устройствах.





Сдвиговый регистр



Обозначение сдвиг. RG

Регистры

Назначение регистров – хранение и преобразование много разрядных двоичных чисел. Регистры наряду со счетчиками и запоминающими устройствами являются наиболее распространенными последовательными устройствами цифровой техники. Регистры обладают большими функциональными возможностями. Они используются в качестве управляющих и запоминающих устройств, генераторов и преобразователей кодов, счетчиков, делителей частоты, узлов временной задержки.

Элементами структуры регистров являются синхронные триггеры Д-типа, либо RS(IK)-типа с динамическим или статическим управлением.

Единичный триггер – простейший регистр (RG) используют в RG цепочке триггеров. Понятие "весовой коэффициент" к разрядам регистра неприменимо в отличие от счетчика, поэтому на условных изображениях нумерация входов и выходов идет подряд (Д1, Д2,Д3 и т.д., а не Д1, Д2, Д4, Д8).

RG в зависимости от функциональных свойств делятся на: накопительные и сдвигающие.

По способу ввода, вывода информации – параллельные, последовательные и комбинированные (//-последовательные и последовательно-//).

По направлению передачи (сдвига) информации – однонаправленные и реверсные.

Регистры памяти – простейший вид регистров – хранят двоичную информацию.

Это набор синхронных триггеров, каждый из которых хранит один разряд двоичного числа. Ввод (запись) и вывод (считывание) производится одновременно во всех разрядах. С приходом очередного тактового импульса происходит обновление информации.

Считывание информации в прямом или обратном (с $\overline{Q}$) коде.

2) Назначение, классификация математических моделей и методы их построения. Проверка адекватности математических моделей.

Задача проектирования на современном этапе заключается в разработке нового объекта, исключающего ошибки не только работоспособного, но и оптимального с точки зрения заданного критерия.

Для решения данной задачи необходимо наличие

1. Варьируемых переменных, т.е. таких, которые можно произвольно менять в некоторых пределах.
2. Критерия, т.е. показателя, с использованием которого сравниваются различные варианты проектных решения, полученных при различных значениях варьируемых переменных
3. Мат. модель, связывающая варьируемые переменные и критерий.

Мат. модель – это система:

- Булевых
- Алгебраических
- Дифференциальных уравнений
- Интегральных уравнений

Назначение мат. модели:

- Управление объектом
- Оптимизация действующих объектов
- Проектирование новых объектов

Классификация мат. моделей:

1. По режиму работы объекта (статики и динамики)
2. По свойствам объекта (сосредоточенные координаты (переменные объекта одинаковы во всех точках) и распределенные координаты(наоборот ☺))
3. Линейные (удовл. принципу суперпозиции)/нелинейные (наоборот ☺)

Методы построения мат. моделей

1. Аналитический

2. Экспериментальный
3. Экспериментально-аналитический (Некоторые параметры аналитически построенной модели уточняются с помощью экспериментов)

3) Алгоритмы сжатия графических данных.

Алгоритмы сжатия изображений бурно развивающаяся область машинной графики. Основной объект приложения – изображение, характеризуется тремя особенностями:

1. изображение требует для хранения гораздо больше памяти, чем текст. Это определяет актуальность алгоритмов архивации графики.
2. человеческое зрение при анализе изображения оперирует контурами и общими переходами цветов и сравнительно нечувствительно к малым изменениям в изображении. Это позволяет создать специальные алгоритмы сжатия ориентированные только на изображения.
3. изображения обладают избыточностью в двух измерениях (по горизонтали и вертикали). Т.о. при создании алгоритма компрессии графики мы используем особенности структуры изображения.

На данный момент известно минимум три семейства алгоритмов, разработанных исключительно для сжатия изображений.

Классы изображений:

1. изображения с небольшим количеством цветов (4-16) и большими областями, заполненными одним цветом, плавные переходы цветов отсутствуют.
2. изображения с плавными переходами цветов.
3. фотореалистические изображения.
4. фотореалистические изображения с наложением деловой графики.

Алгоритмы архивации без потерь**.**

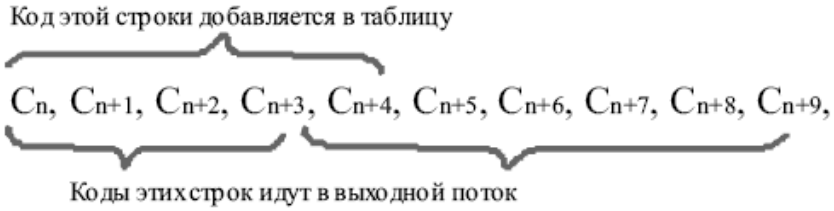
RLE (Run Length Encoding)

Групповое кодирование – самый старый и самый простой алгоритм. Изображение в нем вытягивается в цепочку байт по строкам растра. Цепочки одинаковых байт в изображении заменяются на пары <счетчик повторений, значение>.

LZW (Lempel, Ziv, Welch)

Сжатие происходит за счет одинаковых цепочек байт. Процесс сжатия выглядит достаточно просто. Мы считываем последовательно символы входного потока и проверяем, есть ли в созданной нами таблице строк такая строка. Если строка есть, то считываем следующий символ, а если строки нет, то мы заносим в поток код для предыдущей найденной строки, заносим строку в таблицу и начинаем поиск снова.

Особенность алгоритма заключается в том, что для декомпрессии нам не надо сохранять таблицу строк в файл для распаковки. Алгоритм построен таким образом, что мы в состоянии восстановить таблицу строк, пользуясь только потоком кодов. Мы знаем, что для каждого кода надо добавлять в таблицу строку, состоящую из уже присутствующей там строки и символа, с которого начинается следующая строка в потоке.



Пример:

Пусть мы сжимаем последовательность 45,55,55,151,55,55,55. Тогда мы имеем в выходной строке сначала код очистки <256>. Потом добавим к изначально пустой строке 45 и проверим, есть ли строка 45 в таблице. Поскольку мы при инициализации занесли в таблицу все строки из одного символа, то 45 есть в таблице. Далее читаем следующий символ из входного потока 55 и проверяем есть ли строка 45,55 в таблице. Такой строки в таблице пока нет. Мы заносим в таблицу строку 45,55 с первым свободным кодом 258 и записываем в поток код 45. Можно коротко представить архивацию так:

45 – есть в таблице
45,55 – нет, добавляем в таблицу 258 - 45,55; в поток – 45.
55,55 – нет, в таблицу 259 – 55, 55; в поток – 55.
55,151 – нет, в таблицу 260 – 55,151; в поток – 55.
151,55 – нет, в таблицу 261 – 151,55; в поток – 151.

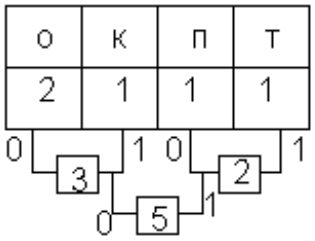
55,55 – есть в таблице.
55,55,55 – нет, в таблицу 262 - 55,55,55; в поток 259.

Алгоритм Хаффмана

Классический алгоритм Хаффмана практически не применяется к изображениям в чистом виде, а используется как один из этапов компрессии в более сложных схемах. Близкая модификация алгоритма используется при сжатии черно-белых изображений. Последовательность подряд идущих черных и белых точек заменяется числом, равным их количеству. А этот ряд сжимается по Хаффману с фиксированной таблицей. Каждая строка изображения сжимается независимо.

Пример:

Сжать по методу Хаффмана *поток*



О – 00
К – 01
П – 10
Т – 11
1000110001

Алгоритмы архивации с потерями

Алгоритм JPEG

Один из самых новых и мощных алгоритмов. Он является стандартом де-факто для полноцветных изображений. Алгоритм оперирует областями 8*8, на которых яркость и цвет меняются достаточно плавно. Вследствие этого, при разложении такой матрицы в двойной ряд по косинусам значимыми являются только первые коэффициенты. Т.о. сжатие осуществляется за счет плавности изменения цветов в изображении. В целом алгоритм основан на применении дискретно-косинусоидального преобразования (ДКП) к матрице изображения для получения новой матрицы коэффициентов. Для получения исходного изображения применяется обратное преобразование.

Алгоритм:

1. переводим изображения из RGB в модель YCrCb (яркость, хроматический красный, хроматический синий). За счет того, что человеческий глаз менее чувствителен к цвету, чем к яркости, появляется возможность архивировать массивы Cr, Cb с большими потерями.

$$\begin{vmatrix} Y \\ Cb \\ Cr \end{vmatrix} = \begin{vmatrix} 0.299 & 0.587 & 0.114 \\ 0.5 & -0.4187 & -0.0813 \\ 0.1687 & -0.3313 & 0.5 \end{vmatrix} * \begin{vmatrix} R \\ G \\ B \end{vmatrix} + \begin{vmatrix} 0 \\ 128 \\ 128 \end{vmatrix}$$

$$\begin{vmatrix} R \\ G \\ B \end{vmatrix} = \begin{vmatrix} 1 & 0 & 1.402 \\ 1 & -0.34414 & -0.71414 \\ 1 & 1.772 & 0 \end{vmatrix} * \left(\begin{vmatrix} Y \\ Cb \\ Cr \end{vmatrix} - \begin{vmatrix} 0 \\ 128 \\ 128 \end{vmatrix} \right)$$

2. разбиваем исходное изображение на матрицы 8*8
3. применяем ДКП к матрицам

$$Y[u,v] = \frac{1}{4} \sum_{i=0}^{i} \sum_{j=0}^{i} C(i,u) \times C(j,v) \times \quad y[i,j]$$

где $C(i,u) = A(u) \times \quad \cos\left(\frac{(2 \times i+1) \times u \times \pi}{2 \cdot n}\right)$

$$A(u) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{for } u \equiv 0 \\ 1, & \text{for } u \neq 0 \end{cases}$$

При этом получаем матрицу, в которой коэффициенты в левом верхнем углу соответствуют низкочастотной составляющей изображения, а в правом нижнем – высокочастотной. Плавное изменение цвета соответствует низкочастотной составляющей, а резкие скачки – высокочастотной.

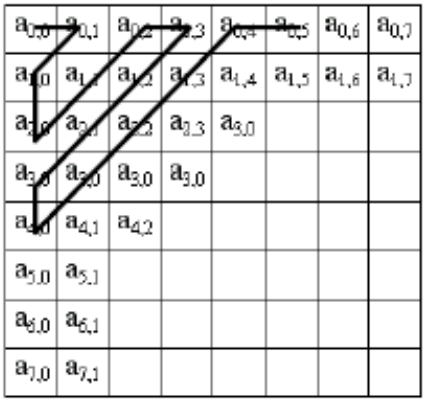
4. квантование – деление рабочей матрицы на матрицу квантования

$$Yq[u,v] = \text{IntegerRound} \left(\frac{Y[u,v]}{q[u,v]} \right)$$

На этом шаге осуществляется управление степенью сжатия и происходят самые большие потери.

5. переводим матрицу 8*8 в 64-элементный вектор при помощи зигзаг-сканирования. Т.о. получаем в начале вектора коэффициенты, соответствующие низким частотам, а в конце – высоким.

6. свертываем вектор с помощью алгоритма группового кодирования
7. свертываем получившиеся пары кодированием по Хаффману.



Билет 24

1) Математические модели процессов теплопереноса.

1. Теплопроводность распространение тепла за счет колебательных движении атомов и молекул. Наблюдается в твердых телах и тонких неподвижных слоях жидкости и газа. Теплопроводность описывается законом Фурье:

$dq = \lambda \frac{\partial T}{\partial x} dF$ **,** где dq – количество тепла, переданное в единицу времени через площадь dF; λ - коэффициент теплопроводности.

Изменение температуры в любой точке объема в любой момент времени можно найти из уравнения теплопроводности вида:

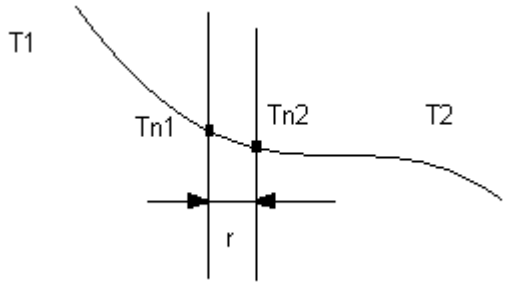
$$\frac{\partial T(\tau,x,y,z)}{\partial \tau} = \frac{\lambda}{c\rho}\left(\frac{\partial^2 T(\tau,x,y,z)}{\partial x^2} + \frac{\partial^2 T(\tau,x,y,z)}{\partial y^2} + \frac{\partial^2 T(\tau,x,y,z)}{\partial z^2}\right)$$

2. Конвективный теплоперенос – тепло передается из-за разности плотностей. Такой способ передачи тепла возможен для жидкостей и газов. Описывается уравнением теплоотдачи:

$dq = \alpha(T1 - T2)dF$

α - коэффициент теплоотдачи

Передача тепла от одной среды к другой через бесконечно плоскую стенку.



Поток тепла от теплоносителя передается стенке вследствие теплопередачи. Этот же поток тепла передается через стенку вследствие теплопроводности. Далее он передается хладоагенту вследствие теплоотдачи

$$dq = \alpha 1(T1 - Tn1)dF$$

$$dq = \frac{\lambda}{r}(Tn1 - Tn2)dF$$

$$dq = \alpha 2(Tn2 - T2)dF$$

$$+\begin{cases} \frac{dq}{\alpha 1 dF} = T1 - Tn1 \\ \frac{dq}{\lambda / r dF} = Tn1 - Tn2 \\ \frac{dq}{\alpha 2 dF} = Tn2 - T2 \end{cases}$$

$$\frac{dq}{dF}\left(\frac{1}{\alpha 1} + \frac{r}{\lambda} + \frac{1}{\alpha 2}\right) = T1 - T2$$

$$dq = kt(T1 - T2)dF$$

kt – коэффициент теплопередачи

3. Излучение – передача тепла электромагнитными волнами, единственный вид теплопереноса, не требующий теплопередающей среды

$dq = kn(T_1^4 - T_2^4)dF$

kn – коэффициент излучения

T1 – температура излучающего тела

T2 – температура принимающего тела

$kn = \frac{1}{\frac{1}{k1} + \frac{1}{k2} - \frac{1}{ka}}$, k_a – для абсолютно черного тела

2) Интерполяционные кривые в машинной графике.

Одной из распространенных задач в САПР является графическое представление кривых и поверхностей. Средства компьютерной графики существенно помогают при проектировании, показывая конструктору различные варианты моделирования поверхности. При этом часто возникает задача построения кривой или поверхности, проходящей через ряд заданных точек.

Кривые Безье

Важное значение при формировании как 2D, так и 3D моделей имеет построение элементарных кривых. Кривые строятся, в основном, следующими способами:

- той или иной интерполяцией по точкам,
- вычислением конических сечений,
- расчетом пересечения поверхностей,
- выполнением преобразования некоторой кривой,
- формированием замкнутых или разомкнутых контуров из отдельных сегментов, например, отрезков прямых, дуг конических сечений или произвольных кривых.

В качестве последних обычно используются параметрические кубические кривые, так как это наименьшая степень при которой обеспечиваются:

- непрерывность значения первой (второй) производной в точках сшивки сегментов кривых,
- возможность задания неплоских кривых.

Параметрическое представление кривых выбирается по целому ряду причин, в том числе потому, что зачастую объекты могут иметь вертикальные касательные. При этом аппроксимация кривой y = f(x) аналитическими функциями была бы невозможной. Кроме того кривые, которые надо представлять, могут быть неплоскими и незамкнутыми. Наконец, параметрическое представление обеспечивает независимость представления от выбора системы координат и соответствует процессу их отображения на устройствах: позиция естественным образом определяется как две функции времени x(t) и y(t).

В общем виде параметрические кубические кривые можно представить в форме:

$$\begin{aligned} x(t) &= A_{11}\,t^3 + A_{12}\,t^2 + A_{13}\,t + A_{14} \\ y(t) &= A_{21}\,t^3 + A_{22}\,t^2 + A_{23}\,t + A_{24} \\ z(t) &= A_{31}\,t^3 + A_{32}\,t^2 + A_{33}\,t + A_{34} \end{aligned} \qquad (0.2.2)$$

где параметр t можно считать изменяющимся в диапазоне от 0 до 1, так как интересуют конечные отрезки.

Существует много методов описания параметрических кубических кривых. К наиболее применяемым относятся:

- метод Безье, широко используемый в интерактивных приложениях; в нем задаются положения конечных точек кривой, а значения первой производной задаются неявно с помощью двух других точек, обычно не лежащих на кривой;

В форме Безье кривая в общем случае задается в виде полинома Бернштейна:

$$P(t) = я\sum_{i=0}^{n} Cm_i\ t^i\ (1-t)^{m-1}\ P_i$$

где P_i - значения координат в вершинах ломаной, используемой в качестве управляющей ломаной для кривой, t - параметр,

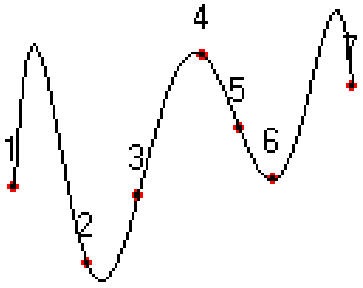
$$Cm_i = \frac{m!}{i!\ (m-i)!}$$

При этом крайние точки управляющей ломаной и кривой совпадают, а наклоны первого и последнего звеньев ломаной совпадают с наклоном кривой в соответствующих точках.

Предложены различные быстрые схемы для вычисления кривой Безье.

Кривые Лагранжа

Пусть на плоскости задан набор точек :



(x_i, y_i), i= 0, 1, ... , m , причем $x_0 < x_1 < x_2 < ... < x_m$.

многочлен Лагранжа - $L(x) = \sum_{i=0}^{n} y_i \prod_{j=0\, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$

Если раскрыть данный многочлен, получим

$$L_m(x) = \frac{(x-x_1)(x-x_2)...(x-x_n)}{(x_0-x_1)(x_0-x_2)...(x_0-x_n)}\,y_0 + \frac{(x-x_0)(x-x_2)...(x-x_n)}{(x_1-x_0)(x_1-x_2)...(x_1-x_n)}\,y_1 + ...$$

Кривые Эрмита:

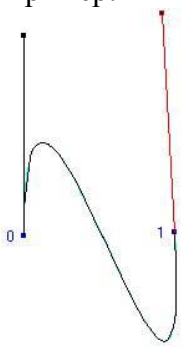
Используются и многие другие методы, например, метод Эрмита, при котором задаются положения конечных точек кривой и значения первой производной в них.

Даны точки, из каждой выходит вектор, являющийся касательной к куску кривой, выходящей из этой точки.
Для построения простейшей кривой достаточно двух точек. Существует параметр t, принадлежащий отрезку [0,1] и изменяющийся на нем с некоторым шагом.

$$x_t = x_1(1+2t^3-3t^2)+x_2(-2t^3+3t^2)+x_{B1}5(t^3-2t^2+t)+x_{B2}(t^3-t^2)$$

$$y_t = y_1(1+2t^3-3t^2)+y_2(-2t^3+3t^2)+y_{B1}5(t^3-2t^2+t)+y_{B2}(t^3-t^2)$$, где x_{B1} , y_{B1} и x_{B2} , y_{B1} - координаты векторов относительно точек.

Пример:



Геометрические сплайны

В некоторых случаях требуется обеспечить глобальную гладкость интерполяционного полинома на всем отрезке.
Сплайн – особым образом построенные гладкие кусочные функции, сочетающие в себе локальную простоту и глобальную для всего отрезка интерполяции гладкость.
B-сплайны:
В более общей форме B-сплайнов кривая в общем случае задается соотношением:

$$P(t) = \sum_{i=0}^{n} P_i \quad N_{im}(t)$$

где P_i - значения координат в вершинах ломаной, используемой в качестве управляющей ломаной для кривой, t - параметр, N_{im} - весовые функции, определяемые рекуррентным соотношением:

$$N_{i,1} = \begin{cases} 1, & \text{если } x_i \le t \le x_{i+1} \\ 0, & \text{в других случаях} \end{cases}$$

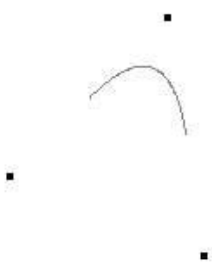
$$N_{i,k}(t) = \frac{(t - x_i)\, N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)\, N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}.$$

Квадратичный:
Для его построения требуется минимум три точки.

Тогда $x_t = t^2 x_3 + (0.5 - t + t^2)x_2 + (0.5 - t^2 + t)x_1$

$$y_t = t^2 y_3 + (0.5 - t + t^2)y_2 + (0.5 - t^2 + t)y_1$$

Пример:



Кубический:
Для его построения требуется минимум 4 точки.

$$x_t = x_1(-\frac{t^3}{3}+t^2+\frac{1}{2}(-t)+\frac{1}{3})+x_2(-t^3-2t^2+\frac{2}{3})+x_3(-t^3+t^2+\frac{t}{2}+\frac{1}{6})+x_4\frac{t^3}{3}$$

$$y_t = y_1(-\frac{t^3}{3}+t^2+\frac{1}{2}(-t)+\frac{1}{3})+y_2(-t^3-2t^2+\frac{2}{3})+y_3(-t^3+t^2+\frac{t}{2}+\frac{1}{6})+y_4\frac{t^3}{3}$$

Пример:

3) Алгоритм определения производительности по тестируемой команде

1. НАЧАЛО
2. Вводим частоту процессора тестируемого компьютера в МГц в переменную f
3. f := f*1000000
4. засекаем текущее время и переводим его в секунды, и сохраняем в переменной T1
5. В цикле от 1 до 1000000000 выполняем тестируемую команду.
6. засекаем время и переводим его в секунды, и сохраняем в переменной T2
7. находим TimeTest1 := T2 – T1
8. засекаем текущее время и переводим его в секунды, и сохраняем в переменной T3
9. Выполняем пустой цикл от 1 до 1000000000 (для определения времени выполнения цикла)
10. засекаем время и переводим его в секунды, и сохраняем в переменной T4
11. находим TimeTest2 := T4 – T3
12. находим число операций выполненных за 1 секунду Oper := 1000000000/(TimeTest1 – TimeTest2)
13. находим число тактов необходимое для выполнения команды: NumTakt := f/Oper
14. КОНЕЦ

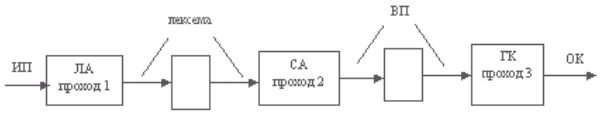
Билет 25
1) Трансляторы. Виды. Состав.

Операционная система содержит комплекс программ, называемых трансляторами. Транслятор обеспечивает автоматический перевод программ с алгоритмического языка в машинные коды.

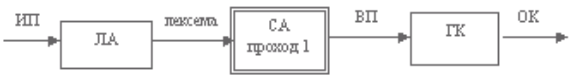
Трансляторы делятся на:

Компиляторы – осуществляют перевод с языка высокого уровня на язык машиных кодов и создают объектную программу.
В состав компилятора входят лексический и синтаксический анализаторы, генератор кода.
На вход лексического анализатора поступает цепочка символов некоторого алфавита, которые он группирует в единые объекты – лексемы (как правило это зависит от определения языка, любая лексема состоит из двух частей – первая определяет тип лексемы, а вторая – информацию о ней).
На вход семантического анализатора поступает цепочка лексем, которые он исследует и определяет, удовлетворяет ли она структурным условиям, сформулированным в определении синтаксиса языка. Здесь исследуется сначала тип лексемы. Также на данном этапе строится внутреннее представление программы, необходимое для генерации кода.
На вход генератора кода поступает внутреннее представление программы, осуществление его перевода в машинный язык.
Компиляторы делятся в зависимости от варианта построения на:

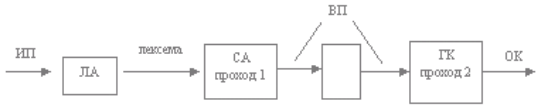
- трехпроходные, недостатком которого является медленная работа. А достоинством – независимость любого элемента, что обеспечивает эффективное использование памяти и возможность оптимизации



- Однопроходный, основной недостаток которого – сложность обработок меток вперед goto



- Двухпроходный



Интерпретаторы – осуществляют перевод программы с языка высокого уровня на язык машинных кодов одновременно с выполнением программы.
Первая часть интерпретатора подобна первой и второй части компилятора, при исполнении программы во внутреннем представлении возникают затруднения, связанные с данными операций. Даже если в языке используются данные одного типа, стек при исполнении внутреннего представления может содержать 3 вида значений – конкретное число (возникают при исполнении операций), указатель на таблицу символов (когда в стек заносятся элементы из постфиксной записи), адрес переменной (если невозможно использовать таблицу символов непосредственно). Каждый элемент стека содержит 2 поля – вид и значение.
Ассемблеры – программы, которые осуществляют перевод с языка низкого уровня на машинный код. Для этого необходимо выполнить следующее:

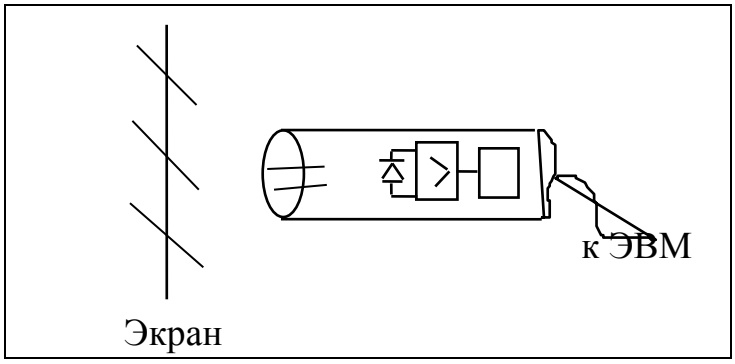
- преобразовать коды операций в их эквиваленты на машинном языке
- преобразовать символьные операнды в соответствующие им машинные адреса
- построить машинные команды в соответствующем формате
- преобразовать константы в исходное представлении во внутреннее машинное представление.
- Записать объектную программу и выдать листинг.

Все указанные действия, за исключением второго могут быть выполнены простой построчной обработкой команд. Сложности возникнут лишь при обработке адресных ссылок вперед.
Потому большинство ассемблеров работает в два прохода.
Во время первого осуществляется назначение адресов для всех предложений исходной программы, запоминание значений (адресов) присвоенных всем меткам исходной программы для последующего использования на втором этапе, выполнение адресных директив.
На втором проходе транслируются команды, генерация данных, заданных адресными директивами, запись объектного кода и листинг.

Любой ассемблер содержит 2 основных внутренних таблицы – таблицу кодов операций (содержит имя операции, соответствующий ей машинный эквивалент, информацию о диске(?) и формате операции) и таблицу символов.
Существуют также однопросмотровые ассемблеры (не могут обрабатывать ссылки вперед), и многопросмотровые.

2) Технические средства диалога машинной графики (световое перо, мышь, шар, джойстик). Конструкция основные характеристики.

1.Световое перо



Внешне имеет вид шариковой ручки или карандаша, соединённого проводом с одним из портов ввода-вывода (или видеоадаптером) компьютера. Ввод данных с помощью светового пера заключается в прикосновениях или проведении линий пером по поверхности экрана монитора, с использованием кнопок, имеющихся на пере, или без такового.

2. Мышь

Мышь - механический манипулятор, преобразующий механические движения в движение курсора на экране.
Мышь воспринимает своё перемещение в рабочей плоскости и передаёт эту информацию компьютеру. Программа, работающая на компьютере, в ответ на перемещение мыши производит на экране действие, отвечающее направлению и расстоянию этого перемещения. В универсальных интерфейсах с помощью мыши пользователь управляет специальным курсором. В дополнение к детектору перемещения мышь имеет от одной до трех (или более) кнопок, а также дополнительные элементы управления (колёса прокрутки, потенциометры, джойстики, трекболы, клавиши и т. п.), действие которых обычно связывается с текущим положением курсора (или составляющих специфического интерфейса).
Виды компьютерных мышей:

Оптомеханические (шариковые) мыши

В оптомеханических (шариковых) мышах шарик с резиновым покрытием 'перекатывается' по поверхности и при своем движении вращает два ролика, отвечающие за перемещение курсора вдоль вертикальной и горизонтальной осей координат.

Оптические мыши первого поколения

Оптические датчики призваны непосредственно отслеживать перемещение рабочей поверхности относительно мыши. Исключение механической составляющей обеспечивало более высокую надёжность и позволяло увеличить разрешающую способность детектора.

Оптические мыши второго поколения

Второе поколение оптических компьютерных мышей имеет более сложное устройство. В нижней части мыши установлен специальный светодиод, который подсвечивает поверхность, по которой перемещается мышь. Миниатюрная камера «фотографирует» поверхность более тысячи раз в секунду, передавая эти данные процессору, который и делает выводы об изменении координат.

Оптические лазерные мыши

В оптических лазерных мышах для подсветки поверхности используется лазер. Лазер, в отличие от светодиода, испускает узконаправленный пучок света, благодаря чему получаемые сенсором изображения более контрастны, а позиционирование курсора достигает высокой точности.

Индукционные мыши

Индукционные мыши используют специальный коврик, работающий по принципу графического планшета или собственно входят в комплект графического планшета. Некоторые планшеты имеют в своем составе манипулятор, похожий на мышь со стеклянным перекрестием, работающий по тому же принципу, однако немного отличающийся реализацией, что позволяет достичь повышенной точности позиционирования за счёт увеличения диаметра чувствительной катушки и вынесения её из устройства в зону видимости пользователя.

Гироскопические мыши

Работа гироскопических мышей основывается на двуосном гироскопическом датчике, который отслеживает перемещения мыши в пространстве. Для работы таких мышей не требуется поверхность, их можно перемещать прямо в воздухе. Подобное решение может оказаться актуальным при недостатке пространства на рабочем столе, а также во время проведения презентаций, когда курсор мыши используется в качестве указки.

3. Шар (трекбол) - в сущности перевернутая мышь. В центре у него большой шар и две клавиши по краям. Вы можете пальцами вращать шар, а нажимать на клавиши большим пальцем. Поскольку он перемещается по столу, а вращается на месте, многие считают его более удобным (особенно для больших экранов) по сравнению с традиционной мышью.

3) Записать алгоритм решения нелинейного уравнения методом Ньютона.

Пусть дано уравнение f(x) = 0.

1. Проверить на сходимость уравнение для решения его методом Ньютона.
2. Если сходится, то задаем начальное приближение x_0, иначе необходимо применять другой метод.
3. Следующие приближения вычисляются по формуле:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

4. Если достигнута заданная точность вычислений $\varepsilon>0$, то переход к п. 5 Иначе переход на пункт 3.
5. Конец.

Билет 26

1) Автоматизация методов управления, вариантного, адаптивного и нового планирования в АСТПП.

1. Управление ТПП.

Метод управления ТПП заключается в организации хранения информации по технологическим маршрутам в соответствии с требованием заказа.
Этот метод применяется в качестве повторного планирования. Его область применения является ограниченной, так как повторяемость обрабатываемых деталей, как правило, невелика.

2.Вариантное планирование.
Исходной предпосылкой данного метода является разбиение инженерами-технологами деталей на классы. В каждый класс входят детали, изготавливающиеся по аналогичной технологии. В каждом классе выделяются детали-представители, которые являются обобщённым представителем, включающим все специфические особенности каждой детали. Для такой детали-представителя разрабатывается стандартный технологический маршрут. Для каждой конкретной детали данного класса выбирается вариант стандартного маршрута, являющегося его подмножеством.
Вариантное планирование предусматривает возможность уточнения стандартного маршрута путём изменения параметров процесса в определённых границах. Увеличение числа обрабатываемых элементов не допускается.
Стандартный вариант R1-R4, D1,D2 - переменные размеры, задав которые получим из стандартного варианта технологические маршруты согласно заданию.
Вариантный метод наиболее употребим на предприятиях с сильно ограниченной номенклатурой деталей. Ограничения на номенклатуру значительно снижают степень гибкости системы ТПП.

3.Адаптивное планирование.
Первым этапом данного метода является построение некоторого множества технологических маршрутов инженерами-технологами. На этапе технологического проектирования осуществляется поиск наиболее близкого к заданному технологического маршрута из имеющихся с помощью определённого классификатора. Далее выбранный технологический маршрут адаптируется к конкретным требованиям заказчика путём добавления, удаления, изменения отдельных шагов проектирования.
Например: Требуется построить технологический маршрут для изготовления детали:
По классификатору определили, что наибольшим близким из имеющихся, является технологический маршрут для изготовления детали.
Для построения технологического маршрута для заданной детали из имеющегося технологического маршрута удаляются операции изгиба пластины в сечении А.
Адаптивное планирование в противоположность методам управления и вариантного планирования обеспечивает порождение дополнительных технологических данных.

4.Метод нового планирования.
Позволяет вести разработку технологических маршрутов для подобных и новых деталей в соответствии с общими и специфическими данными и правилами технологического проектирования. Основой этого служат описания деталей и требования, предъявляемые к её обработке. Анализ этих требований позволяет выявить возможные пути решения технологических задач и в соответствии с определёнными критериями выбрать метод решения. Таким образом, этот метод является и генерирующим. Наиболее ценен в связи с этим и наиболее сложен для автоматизации.
Следует отметить, что границы методов весьма условны. Возможно сочетание отдельных элементов различных методов. Выбор метода для конкретной задачи зависит от условий производства, способов изготовления, назначения изделий, а также от субъективных факторов.

2) Модели гидродинамики

Гидродинамика - описывает движение через аппарат реакционной среды, которая может быть жидкой, газообразной, смесью газовой - жидкой фазы, жидкой - твердой, сыпучей.

1. Модель гидродинамики идеальной смешение:
Примеры объектов описываемые этой моделью:

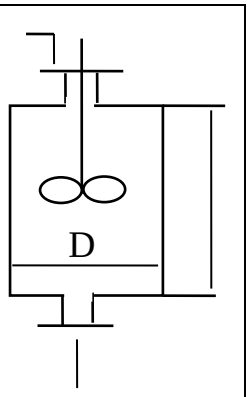


Рис. 1

1) аппараты с мешалкой (Н/Д~=1 почти ид. смешение).
Рисунок 1. Чем больше мощность сообщаемая мешалке тем степень смешения лучше. В зависимости от среды используют разные конструкции мешалок. Чем ниже вязкость среды тем лучше степень перемешивания.

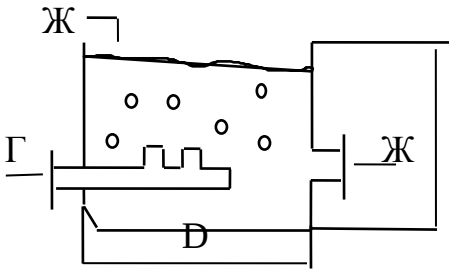


Рисунок 2.

2) аппарат барбатажного типа (пузыри перемешивают поступающую жидкость). Рисунок 2. Применяют в случае когда необходимо провести хим. Реакцию между Г и Ж или провести абсорбцию. Достоинство: большая площадь суммарного контакта Ж и Г. Чем меньше вязкость Ж тем лучше смешение. Чем больше расход газовой фазы тем лучше.

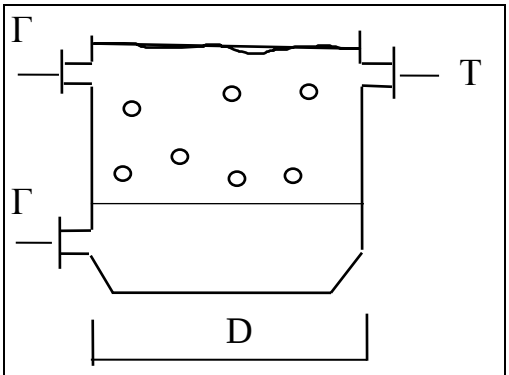


Рисунок 3.

3) Аппарат псевдоожиженного слоя. Рисунок 3. Чем меньше частицы твердой фазы тем лучше смешения. Чем больше расход газа тем лучше.

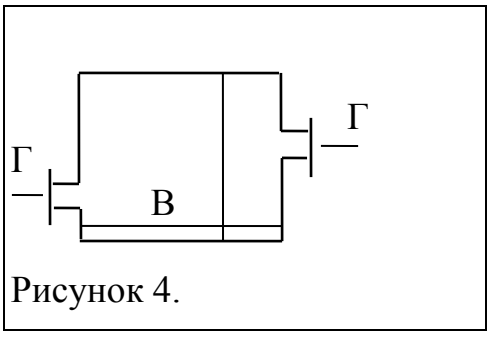


Рисунок 4.

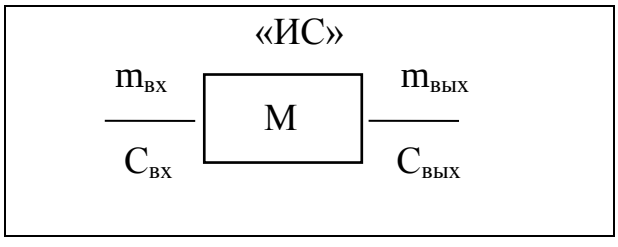
4) реактор полого типа (Рисунок 4.) предназначен для окисление газов. Все фазы - газовые. Смешение происходит под действием диффузии.

Основной характеристикой рассмотренных примеров яв-ся одинаковость в любой точке объема всех хар-к: концентрации, температуры, плотности, вязкости и т.д. Основные допущения модели идеального смешения:

$C(x,y,z.t) = C(t)$ концентрация не зависит от линейных координат(x,y,z)

$T(x,y,z) = T(t)$

С, Т и т.д. в любой точке объектов в том числе и точке выхода одинакова.



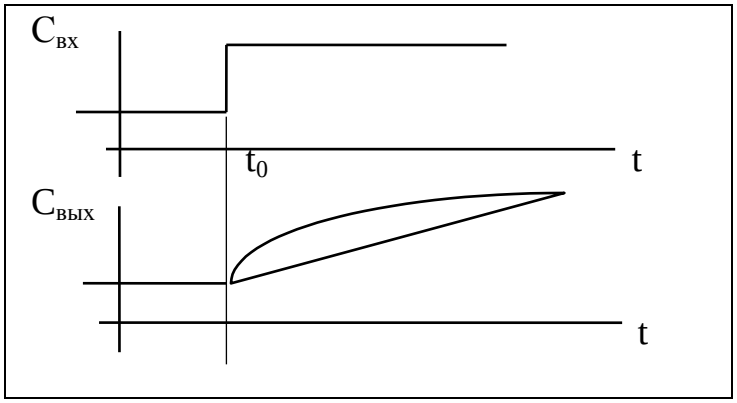
материальный баланс по сухому в-ву:

$$\frac{d(MC_{вых})}{dt} = m_{вх}C_{вх} - m_{вых}C_{вых}$$

Частный случай: $\frac{dM}{dt} = 0$, M=const

$$\frac{dC_{вых}}{dt} = \frac{1}{\bar{t}}(C_{вх} - C_{вых})$$

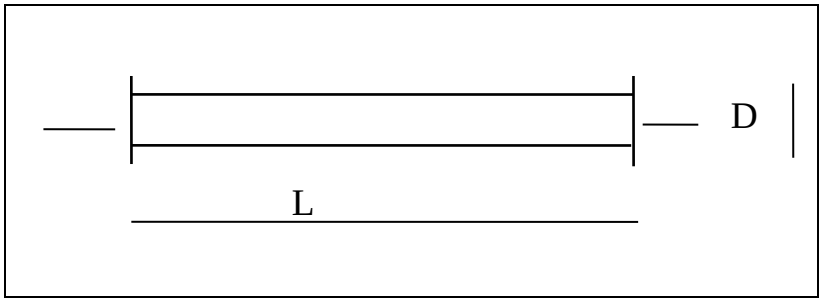
$\bar{t} = \frac{M}{m}$ - среднее время пребывания среды в аппарате.



Преимущества модели ИС: 1) отсутствие параметров определяемых экспериментально; 2) простота расчета.

2. Модель идеальное вытеснение

Аппараты: трубчатые реакторы, трубчатые теплообменники.



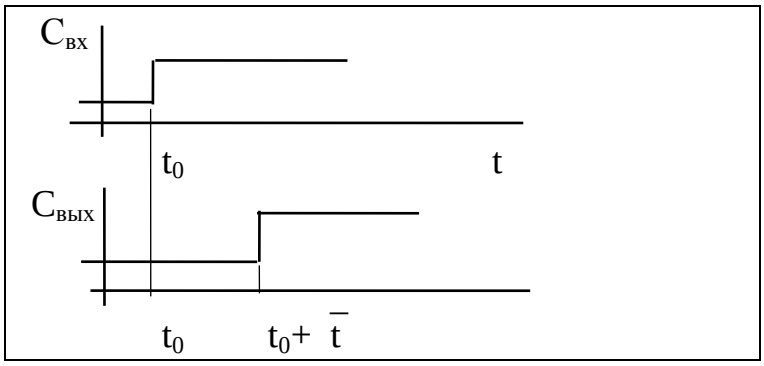
L/D >> 1, обычно L/D ≈ 100.
Основные допущения модели ИВ:
1) среда движется слоями так, что перемешивание последующего слоя с предыдущим отсутствует;
2) среда движется в поршневом режиме, так что перемешивание внутри одного слоя по радиусу отсутствует.
C(x, y, l, t) = C(l, t)

Уравнение описывающее модель ИВ:

$$\frac{\partial C}{\partial t} = -U\frac{\partial C}{\partial l}$$, где

$U = \frac{m}{F\rho}$ - линейная скорость движения, F - поперечного сечения; ρ-плотность среды; m - массовый расход среды.

$\bar{t} = \frac{L}{U}$ - среднее время пребывания среды в объекте.



3. Гидродинамические диффузионные модели.

Ряд объектов характеризуется общим поступательным движением среды от входы к выходу с частичным перемешиванием в продольном и радиальном направлениях.

1) насадочная колонна (рисунок 5)

1< H/D <100

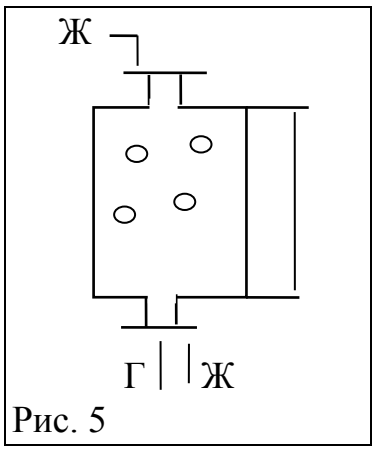
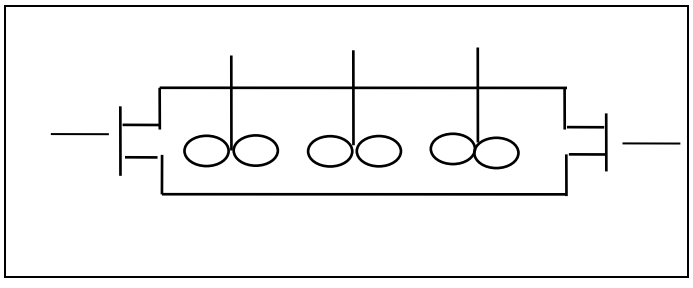


Рис. 5

2) труба с мешалкой



Однопараметрическая гидродинамическая диффузионная модель

$$\frac{\partial C}{\partial t} = -U\frac{\partial C}{\partial l} + D_l\frac{\partial^2 C}{\partial l^2}$$

∂l - коэффициент продольной диффузии

D_l - параметр определяемый экспериментально.

Однопараметрическая модель записана с допущением об отсутствии перемешивания в радиальном направлении.

Двухпараметрическая гидродинамическая диффузионная модель

$$\frac{\partial C}{\partial t} = -U\frac{\partial C}{\partial l} + D_l\frac{\partial^2 C}{\partial l^2} + \frac{D_R}{R}\left(\frac{\partial}{\partial R}\left(R\frac{\partial C}{\partial R}\right)\right)$$

R - радиус аппарата;

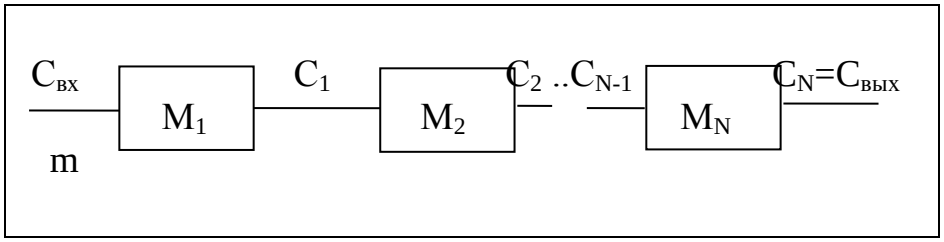
D_R - коэффициент радиальной диффузии, определяется экспериментально.

Диффузионные гидродинамические модели значительно лучше описывают реальные объекты, чем модели ИС и ИВ. Однако ввиду их высокой сложности и необходимости экспериментально определять коэффициенты D_l и D_R , модели получили ограниченное распространение.

4. Гидродинамическая модель ячеечного типа.

В химической технологии распространены объекты представляющие собой каскады реакторов с мешалкой, тарельчатые колонны.

Структурная схема этих объектов:



Допущения:

1) объект представляет собой последовательно соединенные ячейки ИС;
2) между ячейками перемешивание отсутствует;
3) из i-ой ячейки среда переходит только в i+1.

$$\begin{cases} \dfrac{dC_1}{dt} = \dfrac{1}{\bar{t}_1}(C_{вх} - C_1) \\ \dfrac{dC_2}{dt} = \dfrac{1}{\bar{t}_2}(C_1 - C_2) \\ \dfrac{dC_1}{dt} = \dfrac{1}{\bar{t}_T}(C_{N-1} - C_{вых1}) \end{cases}$$

Единственная сложность при использовании модели ЯТ для описания аппаратов колонного типа, это выбор оптимального количества ячеек. При $N\to\infty$ => ИВ, при $N\to 1$ => ИС.

Преимущество модели ЯТ по сравн. С диффузионными моделями заключается в простоте и отсутствии параметров определяемых экспериментально. Количество условных ячеек N подбирается численным экспериментом.

3) Записать алгоритм поиска экстремума функции Розенброка овражным методом.

1) $f(x)=100(x2-x1^2)^2+(1-x1)^2$

1) выбираем начальную точку A0 i=0; x1=A0x; x2=A0y
2) выбираем шаг градиента gr и шаг оврага h, gr<<h
3) вычисляем частные производные

$Px1=-400(x2-x1^2)x1-2(1-x1)$

$Px2=200(x2-x1^2)$;

1) dx1=-Px1*gr;

dx2=-Px2*gr

x1+=dx1;

x2+=dx2;

2) Если 1-(f(x1,x2)-f(x1-dx1, x2-dx2))/f(x1,x2)>= e, то переход к шагу 3
3) x1= x1 + (rand()-0.5)*2*h;

x2= x2 + (rand()-0.5)*2*h;

4) gr=gr/2;

5) пока gr>e переход к 3.

Билет 27.

1) Современные пакеты прикладных программ математического моделирования.

1. 1960 – индивидуальные решения конкретных систем уравнений ММ, для чего составлялась уникальная программа.
2. 1970 – разработка пакетов программ для решения классов задач

MATLAB - решение систем линейных и нелинейных диф. уравнений

GEMEAD - системы обыкновенных дифференциальных уравнений

MATLAB – высокопроизводительный язык для технических расчетов. Он включает в себя вычисления, визуализацию и программирование в удобной среде, где задачи и решения выражаются в форме, близкой к математической. Типичное использование MATLAB – это:

- математические вычисления
- создание алгоритмов
- моделирование
- анализ данных, исследования и визуализация
- научная и инженерная графика
- разработка приложений, включая создание графического интерфейса

MATLAB – интерактивная система, в которой основным элементом данных является массив.

Система MATLAB состоит из пяти основных частей:

1) Язык MATLAB. Это язык матриц и массивов высокого уровня с управлением потоками, функциями, структурами данных, вводом-выводом и особенностями объектно-ориентированного программирования.

2) Среда MATLAB. Это набор инструментов и приспособлений, с которыми работает пользователь. Она включает в себя средства для управления переменными в рабочем пространстве MATLAB, вводом-выводом данных.

3) Управляемая графика. Это графическая система MATLAB, которая включает в себя команды высокого уровня для визуализации двух и трехмерных данных., обработки изображений, анимации.

4) Библиотека математических функций. Это обширная коллекция вычислительных алгоритмов.

5) Программный интерфейс. Это библиотека. Которая позволяет, которая позволяет писать программы на Си и фортране, взаимодействующие с MATLAB.

3. вторая половина 80-х – построение удобного интерфейса пользователя – STAR
4. конец 90-х – интерфейс достиг развития – от пользователя не требуется составления уравнений – ChemCad

2) Реляционная алгебра. Основные операции. Свойства операций.

Реляционная алгебра, определенная Коддом, состоит из 8 операторов, составляющих 2 группы. В первую входят традиционные операции над множествами:

Объединение (union)(возвращает отношение, содержащее все кортежи, которые принадлежат одному из двух определенных отношений или обоим),

Пересечение (intersect)(возвращает отношение, содержащее все кортежи, которые принадлежат одновременно двум определенным отношениям),

Вычитание (-) (minus)(возвращает отношение, содержащее все кортежи, которые принадлежат первому из двух определенных отношений и не принадлежат второму),

Декартово произведение (*) (times)(возвращает отношение, содержащее все кортежи, которые являются сочетанием двух кортежей, принадлежащих соответственно двум определенным отношениям).

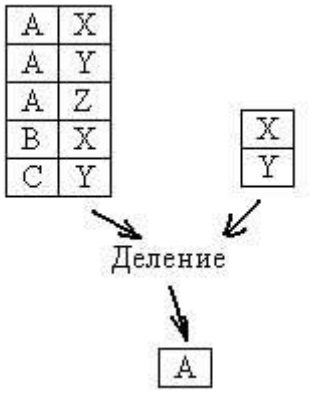
Во вторую группу входят специальные реляционные операции:

Выборка (ограничение) (возвращает отношение, содержащее все кортежи из определенного отношения, которое удовлетворяет определенным условиям.)

Проекция (возвращает отношение, содержащее все кортежи (подкортежи) определенного отношения после исключения из него некоторых атрибутов)

Соединение (возвращает отношение, кортежи которого – это сочетание двух кортежей (принадлежащих соответственно двум определенным), имеющих общее значение для одного или нескольких общих атрибутов этих двух отношений)

Деление (для двух отношений бинарного и унарного, возвращает отношение, содержащее все значения одного атрибута бинарного отношения, которые соответствуют всем значениям в унарном отношении).



Свойство замкнутости – результат каждой операции над отношениями может являться только отношением. Необходимо предусматривать набор правил наследования атрибутов.

Свойства стандартных операций:

- ассоциативность ((A union B) union C ~ A union (B union C) => A union B union C)

-коммуникативность (A union B ~ B union A) (не выполняется для minus)

3) Представить алгоритм метода конечных разностей решения уравнения

$$\frac{\partial^2 F(x,y)}{\partial x^2} = a\frac{\partial^2 F(x,y)}{\partial y^2} + 4xy,$$

$$F(0,y) = 25 - 5y,$$

$$F(x,0) = 15 + \frac{1}{0{,}1x + 0{,}1},$$

$$F(x,10) = 26.6cox$$

$$0 \le x \le 5$$

$$0 \le y \le 10$$

В методе конечных разностей область непрерывного изменения аргумента заменяют конечным множеством точек. Значение функции заменяется значением так называемой сеточной функции, определенной на этом множестве аргументов.

Задание двухточечной краевой задачи имеет вид:

$u''(x)+q(x)u(x)=f(x),\ u(a)=u_a,\ u(b)=u_b$.

Её решение сводится к решению системы алгебраических уравнений вида:

$u_0=u_a$;

$-u_{i-1}+u_i(2+h^2q_i)-u_{i+1}=h^2f_i$;

$u_n=u_b$,

где i от 1 до n-1, h - задаваемый шаг.

Данную систему решают чаще всего методом прогонки, который предназначен для решения систем вида:

$b_0u_0+c_0u_1=d_0$;

$a_iu_{i-1}+b_iu_i+c_iu_{i+1}=d_i$, i от 1 до n-1;

$a_nu_{n-1}+b_nu_n=d_n$.

Метод прогонки состоит из прямого и обратного хода. На прямом ходе вычисляются прогоночные коэффициенты:

$\alpha_0=-c_0/\gamma_0$, $\beta_0=d_0/\gamma_0$, $\gamma_0=b_0$;

$\alpha_i=-c_i/\gamma_i$, $\beta_i=(d_i-a_i\beta_{i-1})/\gamma_i$, $\gamma_i=b_i+a_i\alpha_{i-1}$, i от 1 до n-1;

$\beta_n=(d_n-a_n\beta_{n-1})/\gamma_n$, $\gamma_n=b_n+a_nd_{n-1}$.

Значения функции вычисляются на обратном ходе:

$u_n=\beta_n$;

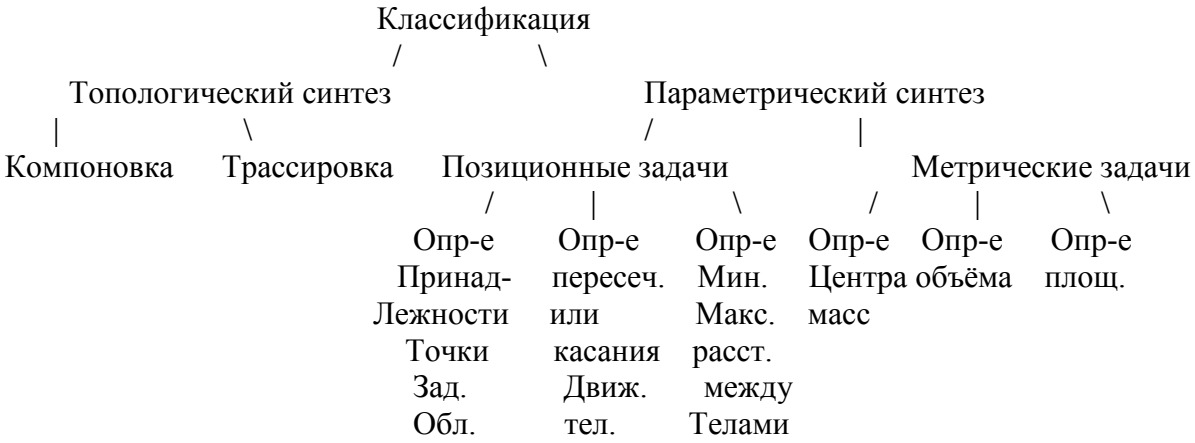
$u_i=\alpha_iu_{i+1}+\beta_i$, i от 0 до n-1.

Решив систему методом прогонки, получают значения функции в узлах сетки.

```
int n=(int)(10.0/h), m=(int)(5.0/dh)+1;
for (i=0; i<n; i++)
{
   F[i][0]=15+1/(0.1*dh*i+0.1);
   F[i][n]=26.6*cos(dh*i);
}
for (i=0; i<m; i++)
   F[0][i]=25-5*y;
for (j=1; j<m-1; j++)
  for (i=0; i<n; i++)
    F[i][j+1]=(a*F[i][j-1]-F[i-1][j]+ F [i+1][j] – dh*dh*4*dh*i*dh*j)/a;
```

Билет 28.

1) Классификация задач конструкторского проектирования.



Понятие топологического проектирования.

На этапе топологического синтеза решаются задачи создания обобщенной структуры конструируемого объекта. При этом решаются задачи компоновки элементов и трассировки связей межу ними.

Компоновка - размещение элементов на плоскости или в объеме, обеспечивающих выполнение технологической машиной заданных функций.

Трассировка – соединение элементов заданными связями.

При решении задач топологического синтеза важнейшая роль отводится конструктору. При этом используются следующие приемы:

1) Метод проб и ошибок
2) Конструктивная преемственность
3) Метод трансформации и инверсии
4) Метод аналогии
5) Метод мозгового штурма

На этапе топологического синтеза две задачи (компоновки и трассировки) успешно решаются с помощью вычислительной техники. Задачи взаимосвязаны и решаются как правило с использованием различных алгоритмов.

2) Реляционная модель данных. Сравнение с иерархической и сетевой моделями.

В реляционной модели (разработана Коддом в 1969-1970 годах) на логическом уровне элемент чаще всего называют атрибутом; используются термины «колонки», «столбец», «поле». Совокупность атрибутов образует кортеж (ряд, запись, строку), а совокупность кортежей – отношение (таблицу).

Связи между файлами устанавливаются динамически в момент обработки данных по равенству значений соответствующих полей. Структуры записей в реляционных БД – линейные.

Каждое отношение имеет ключ, то есть атрибут (простой ключ) или совокупность атрибутов (составной ключ) однозначно идентифицирующий кортеж.

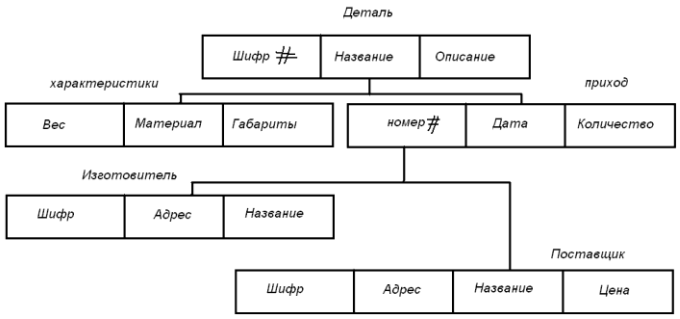
Атрибут или группа атрибутов, не являющаяся ключом в рассматриваемом отношении, а в другом – является, называется внешним ключом.

Если какая-то таблица содержит внешний ключ, то она логически связана с таблицей, содержащей первичный ключ, и эта связь имеет характер один ко многим.

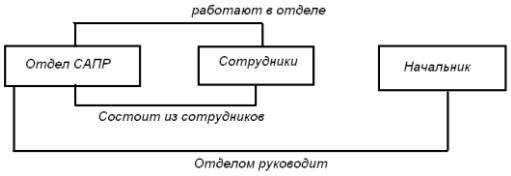
Реляционная модель – где все данные представлены для пользователя в виде прямоугольных таблиц значений данных. Все операции над БД сводятся к манипулированию таблицами, каждая из которых состоит из строк и столбцов, имеет уникальное внутри БД имя.

Пример взаимосвязи таблиц:

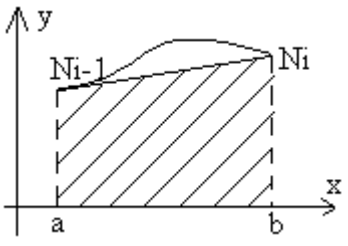
Значения атрибутов выбираются из наименьшей информационной единицы – домена. Домен – множество возможных значений атрибута объекта.
Взаимосвязь таблиц – важнейший элемент реляционной модели. Она поддерживается внешними ключами.
Ограничения целостности требуют, чтобы, например значения атрибутов выбирались только из соответствующего домена, или что внешний ключ не может быть указателем на не существующую строку в таблице.
Различают понятия: переменные отношений и значения отношений. Переменная отношения – обычная переменная - именованный объект, значение которого может изменяться со временем. Значение этой переменной в любой момент времени и будет значением отношения.
В иерархических же моделях имеется один файл – вход в структуру, именуемый корнем дерева. Остальные файлы связаны таким образом, что имеют ровно одну исходную вершину – предка и несколько подчиненных – потомков. ТО есть эта БД состоит из упорядоченного набора нескольких экземпляров одного типа дерева. Этот порядок считается очень важным, поиск осуществляется по составному ключу.
Пример экземпляра:



В сетевых моделях, если не накладывается никаких ограничений, в принципе любой файл может быть точкой входа в систему, и связан с любым числом других файлов, и между записями данных файлов могут быть любые отношения: один к одному, один ко многим, многие ко многим (не поддерживается во многих реальных СУБД). Связи между файлами в сетевой и иерархической модели задаются при создании и передаются с помощью указателей. Пример БД – Integrated Database Management System. Главное отличие от иерархической системы – потомок может иметь любое число предков.
Пример:



3) Написать алгоритм вычисления определенного интеграла методом трапеций.
I = 2 Sinx dx
Формула трапеций. Соединим N_{i-1} (x_{i-1}, f_{i-1}) и N_i(x_i, f_i) на графике функции y=f(x). В результате получится трапеция. Заменим приближенно площадь элементарной криволинейной трапеции площадью построенной фигуры. Получим элементарную квадратурную формулу трапеции: $I \approx \frac{h}{2}(f_{i-1} + f_i)$



Составная квадратурная формула трапеции будет представлять собой:

$$I \approx I_{np}^{n} = h\left[\frac{f_0}{2} + f_1 + f_2 + ... + + f_{n-1} + \frac{f_n}{2}\right] = h\left[\frac{f_0 + f_n}{2} + \sum_{i=1}^{n} f_i\right] (5)$$

Эта формула соответствует замене исходной фигуры ломанной линией, проходящей через точки $N_0,...,N_n$.

```
int a, b, n, s=0, h=0.001;
n=(b-a)/h;
for (i=1; i<=n; i++)
s+=0.5*h*( f((i-1)*h) + f(i*h));
```

Билет 29

1) Задачи автоматизированной системы технологической подготовки производства при использовании станков с ЧПУ.

АСТПП включает в себя решение следующих задач, отсутствующих в ТПП обычных производств:

- автоматизация геометрических расчетов(особенно для сложных поверхностей);
- автоматизация программирования(в простом случае вводится информация о координатах, диаметрах и глубинах отверстий, в сложном случае – диалог с технологом, далее осуществляется синтаксических анализ программы и исправление ошибок технолога, кодирование программы и вывод перфоленты (автоматически));
- графическое моделирование траектории движения инструмента для тестирования программ ЧПУ(выводится на дисплей или графопостроитель, снижает время отладки);

В сложных случаях (обработка одновременно по нескольким направлениям) используется не 3 проекции, а изометрическое представление траектории движения инструмента, осуществляется поворот деталей.

Также существует составление программ сопряжения поверхностей и программ получения сечений.

Этапы:

1) Разработка маршрутной технологии
2) Геометрические расчеты и разработка управляющей программы
3) Подготовка станка к работе и отладка готовой программы на станке

Геометрические расчеты включают в себя описание обрабатываемых поверхностей для целей последовательного программирования, в т.ч. снятие координат и задание базовой и опорных точек..

По сложности:

Расчет перемещений по контуру
Прямолинейных плоских
Криволинейных плоских
Прямолинейных объемных
Криволинейных объемных
Расчет перемещений по ЭКВИДИСТАНТЕ (траектория центра скругляющей дуги)

2) Декомпозиция отношений. Первая, вторая и третья нормальные формы.

Отношение (*таблица*) находится в некоторой нормальной форме, если удовлетворяет заданному условию.

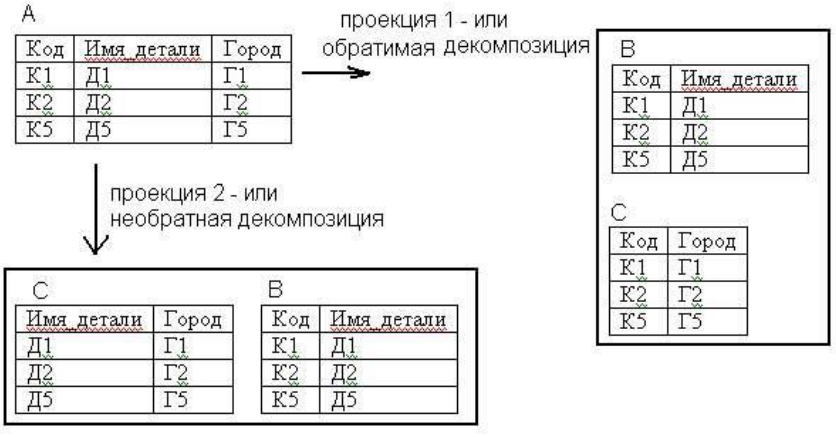
Отношение находится в первой нормальной форме тогда и только тогда, когда оно содержит только скалярные значения. Коддом были определены первая, вторая и третья НФ, вторая НФ более желательна, чем первая и т.д. Бойсом и Коддом переработана 3НФ и в более строгом смысле названа нормальной формой Бойса-Кодда. Есть еще четвертая, определена Фейгином, а так же пятая – проективно-соединительная.

Процедура нормализации включает декомпозицию данного отношения на другие отношения. Декомпозиция должна быть обратимой. Она проводится с помощью теоремы Хеза:

Пусть R{A, B, C} есть отношение, где A, B, C – атрибуты этого отношения. Если R удовлетворяет зависимости A->B, то R равно соединению его проекций {A, B} и {B, C}.

- некоторая функциональная зависимость.

Пример:



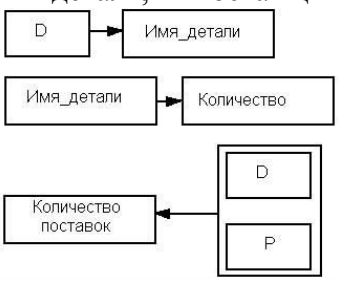
Важную роль играет неприводимая слева функциональная зависимость, например ФЗ {код_детали, код_города, город}может быть записана без атрибута код_города, то есть {код_детали}->город. Последняя ФЗ является неприводимой слева.
Одна из целей проектирования БД – получение НФБК и форм более высокого порядка. Первая, вторая и третья НФ являются промежуточным результатом.

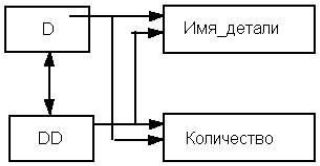
Отношение находится в 1НФ тогда и только тогда, когда все используемые домены содержат только скалярные значения.(каждая ячейка содержит одно значение)

Отношение находится в 2НФ тогда и только тогда, когда оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от первичного ключа. (устраняет столбцы, зависящие от части первичного ключа)

Отношение находится в 3НФ тогда и только тогда, когда оно находится в 2НФ и каждый не ключевой атрибут не транзитивно (то есть отсутствует какая-либо зависимость между столбцами не являющимися первичными ключами) зависит от первичного ключа.

Если в нашем примере, убрать связь между именем детали и количеством, ввести дополнительный независимый атрибут (DD) в качестве потенциального ключа, то получим НФБК.
D – деталь, P- поставщик.





3) Записать алгоритм поиска экстремума функции $f(x_1\ x_2)=x^2_1x_2+(x_2-4)^2$ методом наискорейшего спуска.

1. Ввод функции f (x1, x2) и стартовой точки X^0 (x_1^0, x_2^0)
2. Ввод точности вычислений ε.
3. k=0; // номер итерации
4. Вычисление антиградиента S^k функции f(x1,x2) в точке X^k

$$S^k = \left[-\frac{\partial f(X^k)}{\partial x_1}, -\frac{\partial f(X^k)}{\partial x_2} \right]$$

$$\frac{\partial f(X^k)}{\partial x_1} = \frac{f(x_1+\nabla x, x_2) - f(x_1, x_2)}{\nabla x} \quad // \text{ численный расчет производных}$$

$$\frac{\partial f(X^k)}{\partial x_2} = \frac{f(x_1, x_2+\nabla x) - f(x_1, x_2)}{\nabla x}$$

5. Поиск коэффициента $h^k_{\min}$, из условия, что он доставляет минимум функции $f(X^k + h * S^k)$

Для этого необходимо локализовать отрезок [h_1, h_2] и провести на нем минимизацию любым одномерным методом, например золотым сечением. Локализация отрезка выполняется интуитивным методом.
6. k=k+1;
7. Рассчитываем новую точку X^k

$$X^k = \left(x_1^k = x_1^{k-1} + h_{\min}^{k-1} * S_1^{k-1}; x_2^k = x_2^{k-1} + h_{\min}^{k-1} * S_2^{k-1}\right)$$

8. Рассчитываем критерий остановки. Если

$$\left(\frac{\partial f(X^k)}{\partial x_1} + \frac{\partial f(X^k)}{\partial x_2}\right)^2 \le \varepsilon \text{ , то пункт 9, иначе пункт 4.}$$

9. Конец поиска, точка X^k - доставляет минимум функции f.

Билет 30

1)Метод простой итерации для решения нелинейного уравнения. Понятие сжимающего отображения.

Пусть требуется решить нелинейное уравнение

$$f(x)=0 \qquad (1)$$

Приведем его к виду:

$$x = \varphi(x) \qquad (2)$$

Зададим некоторое начальное приближение x_0 и подставим его в правую часть уравнения (2).

Получим некоторое значение $x_1 = \varphi(x_0)$.

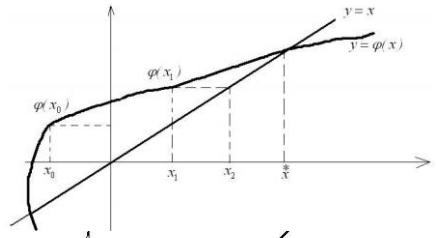
Подставим полученное значение x_1 в правую часть уравнения (2). Проводя этот процесс бесконечно, получим некоторую последовательность.

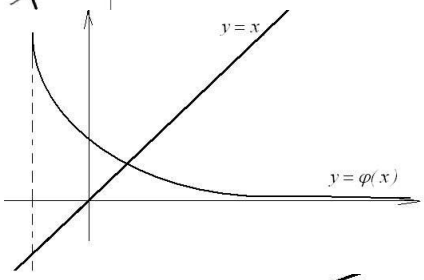
Таким образом, общая формула простой итерации:

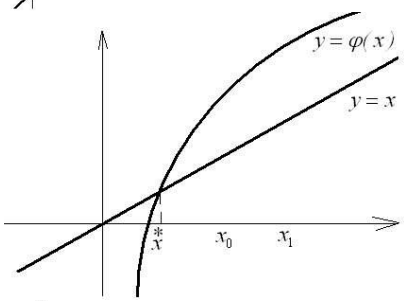
$$x_{k+1} = \varphi(x_k) \quad k \to \infty$$

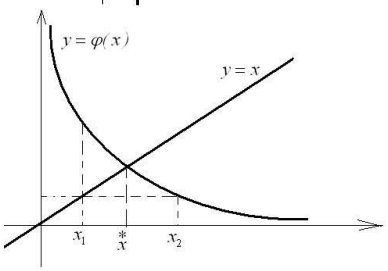
Если для данной последовательности существует предел, то он одновременно является и корнем уравнения (1).

Геометрическая интерпретация:









На первом и втором рисунках взаимное расположение графиков таково, что задача будет решена при произвольном начальном приближении x_0. В третьем и четвертом случаях задача решена не будет, так как независимо от выбора начального приближения итерационный процесс расходится.

Таким образом становится очевидной существенность проверки условия сходимости метода на начальном этапе решения задачи.

Теорема:

Пусть в некоторой $\int$ окрестности корня x^* функция φ дифференцируема и удовлетворяет неравенству:

$0 \le q < 1 \qquad |\varphi'(x)| \le q$, где q – малое число

Тогда независимо от выбора начального приближения x_0 из указанной $\int$ окрестности итерационная последовательность не выходит из нее, и справедливо следующая оценка погрешности:

$|x_n - x^*| \le q^n |x_0 - x^*|$

Обычно для окончания итерационного процесса используется формула:

$|x_i - x_{i-1}| \le \varepsilon$

Сжимающее отображение.

Понятие сжимающего отображения позволяет решить вопрос о сходимости метода аналитически, не прибегая к геометрическому построению.

Рассмотрим некоторую функцию $\varphi(x)$, заданную на отрезке $[a,b]$ и непрерывную на нем.

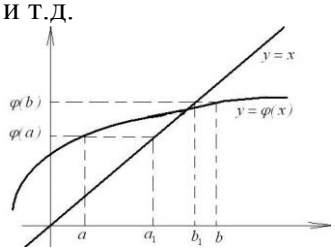
Каждой точке x_0, принадлежащей отрезку $[a,b]$, соответствует некоторое значение $y_0 = \varphi(x_0)$ на оси ординат. То есть $\varphi(x)$ задает отображение отрезка $[a,b]$ на оси ординат.

Построим проекцию отрезка $[\varphi(a),\varphi(b)]$ на ось абсцисс относительно прямой $y = x$. Получим $[a_1,b_1]$.

Если отрезок $[a_1,b_1]$ является частью исходного отрезка $[a,b]$, то функция $\varphi(x)$ отображает отрезок $[a,b]$ в себя.

Произведем этот процесс несколько раз, в результате получим последовательность отрезков $[a,b]$, $[a_1,b_1]$, $[a_2,b_2]$.

и т.д.



Если после каждого отображения исходный отрезок уменьшается в m раз, где m>1, то полученное отображение называется сжимающим.

Таким образом, условие сжатия можно сформулировать так:

Отображение $\varphi(x)$ является сжимающим на отрезке $[a,b]$, если существует такое α, что $0 < \alpha < 1$, для которого выполняется, что для любых двух точек x_1, x_2 выполняется следующее условие:

$$|\varphi(x_1) - \varphi(x_2)| \le \alpha |x_1 - x_2| \qquad \alpha = \frac{1}{m}$$

Разделив обе части полученного неравенства на $|x_1 - x_2|$ и взяв предел от обоих частей данного неравенства, при $|x_1 - x_2| \to 0$ получим формулу:

$$|\varphi'(x)| < \alpha < 1$$

Доказывающую исходную теорему.

Таким образом, можно оценить сходимость функции $\varphi(x)$ для любой произвольной точки x_0.

2) Декомпозиция отношений. Первая, вторая и третья нормальные формы.

Отношение (*таблица*) находится в некоторой нормальной форме, если удовлетворяет заданному условию.

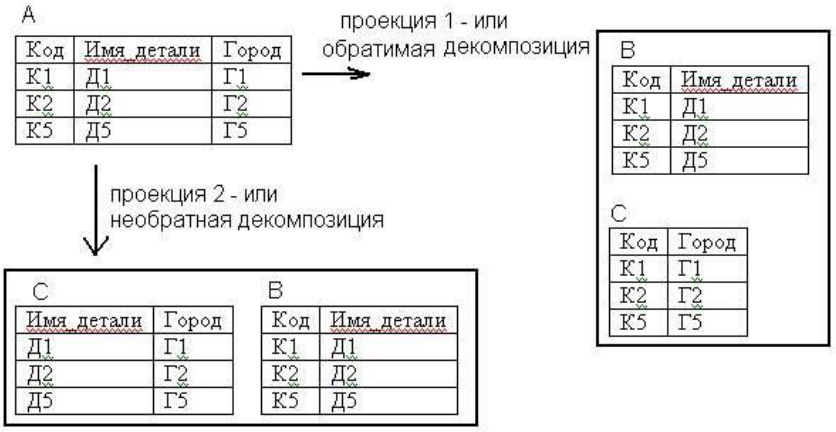
Отношение находится в первой нормальной форме тогда и только тогда, когда оно содержит только скалярные значения. Коддом были определены первая, вторая и третья НФ, вторая НФ более желательна, чем первая и т.д. Бойсом и Коддом переработана 3НФ и в более строгом смысле названа нормальной формой Бойса-Кодда. Есть еще четвертая, определена Фейгином, а так же пятая – проективно-соединительная.

Процедура нормализации включает декомпозицию данного отношения на другие отношения. Декомпозиция должна быть обратимой. Она проводится с помощью теоремы Хеза:

Пусть R{A, B, C} есть отношение, где A, B, C – атрибуты этого отношения. Если R удовлетворяет зависимости A->B, то R равно соединению его проекций {A, B} и {B, C}.

- некоторая функциональная зависимость.

Пример:



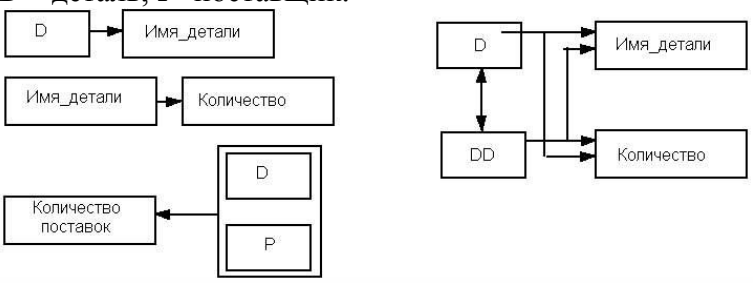
Важную роль играет неприводимая слева функциональная зависимость, например ФЗ {код_детали, код_города, город}может быть записана без атрибута код_города, то есть {код_детали}->город. Последняя ФЗ является неприводимой слева.
Одна из целей проектирования БД – получение НФБК и форм более высокого порядка. Первая, вторая и третья НФ являются промежуточным результатом.

Отношение находится в 1НФ тогда и только тогда, когда все используемые домены содержат только скалярные значения.(каждая ячейка содержит одно значение)

Отношение находится в 2НФ тогда и только тогда, когда оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от первичного ключа. (устраняет столбцы, зависящие от части первичного ключа)

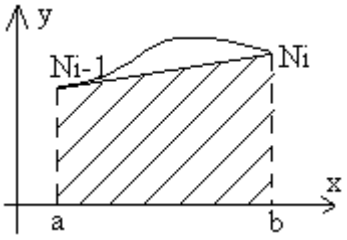
Отношение находится в 3НФ тогда и только тогда, когда оно находится в 2НФ и каждый не ключевой атрибут не транзитивно (то есть отсутствует какая-либо зависимость между столбцами не являющимися первичными ключами) зависит от первичного ключа.

Если в нашем примере, убрать связь между именем детали и количеством, ввести дополнительный независимый атрибут (DD) в качестве потенциального ключа, то получим НФБК.
D – деталь, P- поставщик.



3) Написать алгоритм вычисления определенного интеграла методом трапеций.

$$I = \int_1^2 x^3 \cos x \, dx$$



Формула трапеций. Соединим N_{i-1} (x_{i-1}, f_{i-1}) и $N_i(x_i, f_i)$ на графике функции y=f(x). В результате получится трапеция. Заменим приближенно площадь элементарной криволинейной трапеции площадью построенной фигуры. Получим элементарную квадратурную формулу трапеции: $I \approx \frac{h}{2}(f_{i-1} + f_i)$. Составная квадратурная формула трапеции будет представлять собой:

$$I \approx I_{np}^n = h\left[\frac{f_0}{2} + f_1 + f_2 + \ldots + + f_{n-1} + \frac{f_n}{2}\right] = h\left[\frac{f_0 + f_n}{2} + \sum_{i=1}^{n} f_i\right] (5)$$

Эта формула соответствует замене исходной фигуры ломанной линией, проходящей через точки $N_0,\ldots,N_n$.

```
int a, b, n, s=0, h=0.001;
```

```
n=(b-a)/h;
for (i=1; i<=n; i++)
s+=0.5*h*( f((i-1)*h) + f(i*h));
```

Билет 31

1) Автоматизация задач топологического синтеза при конструировании новых объектов.

Этап создания нового объекта в технике:

1. Предпроектные научно-исследовательские и опытно-конструкторские работы.
2. Конструирование (проектирование) – формируется концепция построения нового объекта и его воплощения в чертежах и текстовых документах. Данный этап отвечает на вопрос, что из себя представляет создаваемый объект.
3. Технологическая подготовка производства. Этап посвящен проектированию технологических процессов изготовления объекта спроектированного на предыдущем этапе. Отвечает на вопрос, как изготавливается изделие.
4. Изготовление объекта
5. Пуск в эксплуатацию.

Основные этапы конструирования нового объекта

1. Анализ назначения технологических процессов, находящихся в новом объекте.
2. Этап топологического проектирования, составление структурной схемы проектируемого объекта (компоновка, трассировка)
3. Параметрический синтез
4. Оптимизация режимов работы объектов
5. Расчет на прочность
6. Силовые расчеты
7. Подготовка и оформление проектной документации

На этапе топологического синтеза решаются задачи создания обобщенной структуры конструируемого объекта. При этом решаются задачи компоновки элементов и трассировки связей межу ними.

Компоновка - размещение элементов на плоскости или в объеме, обеспечивающих выполнение технологической машиной заданных функций.

Трассировка – соединение элементов заданными связями.

При решении задач топологического синтеза важнейшая роль отводится конструктору. При этом используются следующие приемы:

1) Метод проб и ошибок
2) Конструктивная преемственность
3) Метод трансформации и инверсии
4) Метод аналогии
5) Метод мозгового штурма

Автоматизация метода нового планирования наиболее трудоемка, так как при его использовании осуществляется проектирование и документирование ТП на основе введенных данных. ПО исходным данным (описанию детали и программе выпуска) осуществляется выбор заготовки , построение технологического маршрута, выбор оборудования, осуществляются временные расчеты, выбор инструмента, оптимизация проектирования сборочных процессов. Это все отдельные задачи метода нового планирования.

2) Пакет прикладных программ ChemCAD.

Программа ChemCad представляет собой инструментальные средства моделирования химико-технологических процессов для решения задач исследования и проектирования химико-технологических систем, в том числе отдельных аппаратов.

ChemCad имеет модульную структуру и состоит из системного и функционального наполнений, представляющих собой средства и объекты расчета, а также баз данных и интерфейса пользователя, обладающего мощными графическими возможностями.

Моделирование новой технологической схемы с помощью ChemCad’a предполагает следующие этапы:

1. Создать новый файл технологической схемы.
2. Выбрать технические размерности.
3. Выбрать компоненты.
4. Выбрать термодинамические модели.
5. Построить технологическую схему.
6. Задать параметры входных потоков.

7. Задать параметры для всех единиц оборудования.
8. Запустить программу моделирования.
9. Просмотреть результаты моделирования на экране.
10. Распечатать результаты моделирования на принтере.

Эти этапы не обязательно выполнять в такой же последовательности, не обязательно также проходить через все эти этапы при построении технологической схемы, так как для некоторых из них существует информация по умолчанию; но все эти этапы, по крайней мере, следует принять во внимание при решении каждой задачи.

1. Создание нового файла технологической схемы

Для открытия нового задания используется команда **File/New Job (Файл/Новое задание)** на панели инструментов, после чего программа в окне **Сохранение файла** предложит ввести имя файла.

Задание из существующего на диске файла можно открыть, используя команду **File/Open Job... (Файл/Открыть задание...)** на панели инструментов.

После открытия нового задания в заголовке окна выводится его имя, отображаются меню, панель инструментов и **Main Palette (Основная палитра).** Текущий режим программы указывается в строке состояния: **Mode: Flowsheet (Режим: Технологическая схема).** После открытия существующего на диске задания текущим режимом программы является режим **Mode: Simulation (Режим: Моделирование).**

2. Выбор технических размерностей

При создании технологической схемы необходимо выбрать технические размерности. В программе представлены три набора единиц измерения: английский, метрический и СИ. Эти наборы называются профилями единиц измерения.

Для выбора технических размерностей используется команда **Format/Engineering Units (Формат/Единицы измерения).** На экран выводится окно **Engineering Unit Selection (Выбор единиц измерения).**

3. Выбор компонентов

В соответствии с этапами моделирования следующим шагом является задание списка химических компонентов процесса. Выбор компонентов производится из банка данных программы. Для этого используется команда **Ther-moPhisical/Component List (Термофизика/Список компонентов)** на панели инструментов. После выполнения команды на экран выводится окно **Component Selection (Выбрать компонент).** Команда и кнопка доступны в режиме **Mode: Simulation (Режим: Моделирование),** для перехода в который используется команда операционного меню **Edit Flowsheet (Редактирование технологической схемы)** на панели инструментов.

В области **Component Databank (Банк данных компонентов)** перечислены все компоненты всех баз данных системы и локальных пользовательских баз данных. Список компонентов составлен по возрастанию их идентификационных номеров (ID).

4. Выбор термодинамических моделей

Термодинамические свойства потоков определяются заданием любых двух параметров из следующих: температура, давление, доля пара и энтальпия.

Чтобы получить точные результаты расчетов, необходимо выбрать метод, наиболее подходящий для данной химической системы. Выбор термодинамических моделей сводится преимущественно к выбору пригодных методов расчета констант фазового равновесия, энтальпии, энтропии, плотности, вязкости, теплопроводности и поверхностного натяжения содержимого потока. ChemCad содержит примерно 50 методов расчета констант фазового равновесия с различными вариантами и около 12 способов расчета энтальпии. Для выбора термодинамических методов используются команды меню **ThermoPhisical (Термофизика)**, доступные в режиме **Mode: Simulation (Режим: Моделирование)**.

5. Построение технологической схемы

Построение технологической схемы сводится, в основном, к размещению изображений технологического оборудования (далее аппаратов или пиктограмм аппаратов) на экране и соединению их потоками. Иногда на этапе построения схемы возникает необходимость в создании новых и модификаций имеющихся пиктограмм.

6. Изображение потоков на технологической схеме

После завершения размещения аппаратов технологической схемы необходимо соединить их материальными потоками. При изображении потоков следует руководствоваться рядом общих правил.

а). Каждый поток направлен от аппарата-источника к аппарату-приемнику.

б). Каждый аппарат имеет позиции входа и выхода. Они останавливаются при создании пиктограммы аппарата. Программа ориентирует потоки по отношению к этим позициям. Поток всегда направлен из выхода аппарата-источника к входу аппарата-приемника.

в). Начало потока определяется появлением курсора в виде стрелки рядом с позицией выхода из аппарата-источника. При нажатой левой кнопке мыши программа строит поток из этой позиции.
г). При изображении потока, приближаясь к позиции входа аппарата, вновь появляется курсор в виде стрелки. Поток фиксируется нажатием левой клавиши мыши. Одновременно рядом с потоком отображается его ID номер.
д). Для отказа от изображения потока надо нажать правую кнопку и выполнить **Stop drawing stream (Приостановить изображение потока**). Соединение аппаратов потоками выполняется в режиме **Mode: FlowSheet.** В **Main Pallete (Основной палитре)** надо выбрать символ **Stream (Поток),** который позволит указать начало и конец потока.

7. Задание параметров потоков питания и разрываемых потоков

Следующим этапом является задание параметров потоков питания и разрываемых потоков. Термодинамическое состояние потока определяется любыми двумя параметрами из трех следующих: температуры, давления и долей пара; обычно задаются температура и давление. При задании всех трех параметров ChemCad выводит сообщение об избыточном определении потока. Для каждого потока питания нужно задать расход по всем веществам, включенным в список компонентов, либо задаться суммарным расходом компонентов и их концентрациями.
Задание параметров потока можно выполнить следующими способами: дважды щелкнуть левой клавишей мыши на интересующем потоке; использовать команду контекстного меню **Edit Unit Op Streams (Редактирование потоков единицы оборудования)** для задания параметров потоков выбранной единицы оборудования; с помощью команд меню **Specifications (Спецификации).** Задание параметров потоков выполняется в режиме **Mode: Simulation.** Рассмотрим команды меню **Specifications.**

8. Ввод параметров оборудования

По аналогии с заданием параметров потока, для ввода параметров оборудования также используются: двойной щелчок левой клавишей мыши на единице оборудования, команда контекстного меню **Edit Unit Op Data (Редактирование параметров единицы оборудования)** и соответствующие команды меню **Specifications (Спецификации).** Задание параметров оборудования выполняется в режиме **Mode: Simulation.**

9. Запуск программы моделирования

Для проведения моделирования технологической схемы используются команды меню **Run** (**Счет**). С помощью этих команд можно задавать последовательность расчета и выполнять контроль над ходом расчета.

10. Составление отчета

ChemCad позволяет создавать отчет о результатах моделирования в виде таблиц. Их можно вывести на экран, сохранить в текстовом файле со стандартной кодировкой символов (ASCII), в файле типа (PRN) или послать отчет на устройство печати. Программа имеет стандартный формат вывода отчета, однако при необходимости его можно изменить. Можно указать, какие части отчета, а также какие потоки и свойства будут включены в отчет. Имеются опции для задания формата выводимых чисел.

3) Составить алгоритм поиска экстремума функции двух переменных методом покоординатного спуска $F(x_1, x_2) = x_1x_2 + x^2_1x_2 + x_1x^2_2$

Смысл метода в фиксировании одной из переменных и изменении другой, после нахождения первой оптимальной переменной, она фиксируется и начинает меняться другая, пока не находится оптимальное решение.

1) Выбираем X0(x10,x20)-произвольно, e=0.00001,sigma=0.001
2) Фиксируем x2=const
3) x1n+1=x1n+sigma
4) Если f(x1n+1,x2)<f(x1n,x2), то шаг 3 иначе шаг 5
5) Фиксируем x1=const
6) X2n+1=x2n+sigma
7) Если f(x1, x2n+1)<f(x1, x2n), то шаг 6 иначе шаг 8
8) Если ((x1к+1- x1к)2+(x2к+1- x2к)2)>e то шаг 2 иначе шаг 9
9) Вывод x1*, x2*