Дисклеймер.

Автор не несет ответственности за любой ущерб, причиненный Вам при использовании данного документа. Автор напоминает, что данный документ может содержать ошибки и опечатки, недостоверную и/или непроверенную информацию. Если Вы желаете помочь в развитии проекта или сообщить об ошибке/опечатке/неточности:

GitHub проекта

Автор в ВК

Внимание: данный документ не поддерживается и поддерживаться не будет! Сообщения об ошибках НЕ рассматриваются, пулл реквесты НЕ принимаются! Если Вы хотите поддерживать этот документ - форкните проект на Github. Благодарю за понимание.

Последовательности выполнения

a = 1, b = 2, a+b — последовательность.

Fortran — 1957;
ALGOL60 — 1958, 60, 64;
PL1 — 1962-64;
ALGOL68 — 1968-1974;
PASCAL — 1972;
C — 1972, 1978; (связан с разработкой UNIX)
Ada — 1974, 1976;
LISP — 1958;

Условие:
if (a>b)
операторы;
else
операторы;

Циклы:
1)
while (a>b)
{
операторы;
}

2)
do
{
операторы;
}
while (a>b);

float M[100]; - элементы от 0 до 99;
for (i = 0; i < 100; i++)
{
операторы;
}

$a+ = b = \oplus$ — разобраться;

Переменные цикла: i, j, k, l, m.
Конец второй лекции.

Операторы управления C

if, switch - ну, короче, все знают синтаксис.
Лексема - единица транслятора. 135 - 3 литеры, одна лексема. -135 - 4 литеры, 2 лексемы (- и 135).

break работает также в циклах, прерывая их.
continue запускает следующую итерацию, игнорируя последующие команды.

Декомпозиция — разделение сложных структур на более простые.
/* комментарий */ — комментарий.
Юнит-тест (unit test) — проверка отдельного модуля.
Функция void называется процедурой.
Допустимо приведение целого к вещественному, но не наоборот.

Указатели

см. 2.cpp
Функции - см. 3.cpp

Структуры

ad hoc — «для этой цели» статика — (относительно бесплатная) динамика — (дорогая)

Машина Тьюринга

Машина Тьюринга - это бесконечная лента. Имеется головка, которая смотрит на одну из ячеек ленты. И есть конечный алфавит $\{a - z\}$ и пустая ячейка. Также есть конечный набор состояний $\{S_0, ...S_k\}$

Конец второй лекции.

Функции. Повторение

Возможна передача функции нескольких параметров.

Для изменения значения внешней переменной требуется использовать передачу параметров по ссылке. (передачу ссылки в качестве параметра).

Передача массива в качестве параметра функции — см. 5.cpp

По мере возможности сохраняйте все переменные как локальные — (и это называется инкапсуляция)

Понятие сложности алгоритма.

Сложность алгоритма поиска наибольшего элемента массива M[n] — n;

Сложность алгоритма сортировки — n^2;

Сложность алгоритма сортировки пузырьком — $\frac{n^2}{2}$;

Сложность алгоритма сортировки слиянием — $n \log_2 n$

Разобьем массив из 1000 элементов на 256 списков. Каждый из таких списков будет иметь одинаковый хеш $x \to hash(x)$

Бэктрекинг.

Задача представляется в виде дерева решений.

Спускаемся по узлам дерева до получения фэйла. Если фэйл получен, поднимаемся на один узел выше и проверяем другую ветку решений. В случае ошибки - повторить :) Но это решение говенное.(с)Терехов

Конец третьей лекции

Строки

Строка вида "abc" представляется в памяти компьютера лексеммы 4 символов: a, b, c, "конец строки". Адрес строки — адрес первого элемента строки.

Кодировки:
1) Основная с англ. символами: ASCII;
2) Общепринятая в Linux и MacOS: UTF-8 (UTF-16);
3) В Windows: CP1251.

UTF-8 состоит из 8 битов. Если 1 бит равен 0, то кодировка аналогична ASCII. Если 1 бит равен 1, а второй — 0, то добавляется еще один байт, содержащий коды региональных символов.
a[2:5] — взятие подстроки.

Поиск подстроки в строке:
Конечный автомат или матрица состояний.

Данные в памяти компьютера. Биты, байты, слова.

В 1969-1970 годах самой крупной IT-корпорацией была компания IBM с оборотом $18 млрд.
Вот она и ввела следующие стандарты:
байт — 8 bit.
H — 16 bit (полуслово) — всегда кончается 0
W — 32 bit (слово) — кончается 00
DW — 64 bit (двойное слово) — кончается 000.

Представление числовых данных и системы счисления

$1000_2 = 8_{10}$ — прямой код.
$1001_2 = -1_{10}$ — отрицательный элемент в прямом коде. Ясен перец, его никто не юзает, ибо неудобно отличать положительный элемент от отрицательного.
1111 — дополнительный код. Здесь первый бит играет роль знака. Таким образом, вычитание $a - b$ можно осуществлять, переводя b в обратный код и складывая данные переменные.
Перевод из прямого кода в дополнительный: инвертировать прямой код и прибавить единицу.

Представление чисел с плавающей точкой:
Два стандарта: для 32 бит и 64 бит.
Разберем 32 битный стандарт.
Нулевой бит — знак. С 1 по 8 — порядок. С 9 по 31 — мантисса.
Представление: 1.мантисса $* 2^{\text{порядок}}$.

Для 64 бит:
Нулевой бит — знак. 1, 2, 3 биты фиксированы. С 4 по 9 — порядок. Все остальное — мантисса.

Нормализованное число: $\pm exp > 0$ и затем любой набор битов.
Ненормализованное число: ± 0 и затем любые символы не равные 0.
Ноль: ± 0 и нули.
∞: $\pm 1111...$ и затем нули.
NaN (Not a Number): $\pm 111...1$ и затем ненулевые символы.

Конец пятой лекции

Операционные системы. Обзор.

COCOM — ограничение на поставку американцами в СССР электронно-вычислительной техники.

Но наши все равно умудрились своровать IBM/360 и стали выпускать его под названием ЕС ЭВМ.

Потом еще были:
1370 — РЯД 1;
HP — СМ 1, СМ2;
PDPII — СМ3, СМ4;

Но в этот момент англичане вышли с предложением продать нам ICL1900. Но увы, СССР заблокировала эту инициативу.

Каким-то образом все же две ICL1900 с полной документацией попали в СССР.

Инженеры долго не могли разобраться в этой документации из-за проблем с терминологией. Но в конце концов плюнули и стали именовать все термины по-английски.

Функции операционной системы:

1) Планировщик. Он должен понять, сколько задач, сколько каждая требует процессорного времени и раскидать так, как нужно.

2) Управление памятью: MFT — фиксированный захват памяти, MVT — динамический. Каждой задаче нужно какое-то число памяти и ОС должна её выделять.

3) Управление временем.

4) Управление вводом-выводом (одна из главнейших функций). Процессор посылает в канал (channel) команду SIO, канал читает с носителя, процессор в это время работает над другой задачей. Закончив чтение, channel посылает процессору прерывание I/O и он продолжает работу с полученными данными.

Допустим, в памяти два блока. Читаем с первого блока и в это время нихрена не делаешь. Закончив чтение, ты начинаешь работать с полученными данными, а в это время происходит чтение из другого блока и т. д. Два буфера. В один вводятся данные, из другого выводятся. Это называется буферизацией (BSAM).

Когда мы печатаем printf(block), он передается spooling'у. Writer постоянно опрашивает spooling и при появлении там данных начинает их запись на диск.

5) Системы реального времени:

Монитор — часть ОС.

pic — программа управления чем-либо мелким, например, железнодорожной стрелкой. Её задача всего лишь снять данные с датчиков и в зависимости от них перевести стрелку.

Операционные системы бывают:

1) однопрограммные и многопрограммные (в данный момент может быть активна одна или много программ).

2) однопользовательские и многопользовательские (первая предоставляет интерфейс лишь одному пользователю, а вторая способна обрабатывать несколько пользовательских сеансов таким образом, чтобы каждый пользователь был уверен, что единственный работает за данной машиной).

Введение в распределенные вычисления

Распределенное вычисление — вычисление какой-либо сложной программы посредством её разбиения

на несколько малых кусков и их вычисление на нескольких разных машинах.

$\triangle U = \frac{\partial U}{\partial t}\ \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}$

Чтобы такую дуру решить, нужно решить много линейных уравнений Лапласа $\triangle U = \varnothing, t = 0, \triangle t, 2\triangle t, ...$

Распараллеливание процессов имеет сложность $Const\frac{\text{сложность зад}}{\text{число процес}}$

Мужик из Оксфорда McColl придумал систему BSP для вычисления перемножения матриц, такую, что Const = 1.

Конец шестой лекции

Есть такая операционная система RTEMS — для ракет. Абсолютно свободная ОС.

Человеко-машинное взаимодействие.

Первыми интерфейсами взаимодействия были пультами с большим числом лампочек, которыми выводились двоичные данные. Также была печать результатов при помощи перфолент

Потом появились широкие устройства печати наподобие пишущих машинок, имевшие 28 литер.

Постепенно пользовательский интерфейс обогащался, поскольку разрабы начали понимать, что не человек для машины, а машина для человека. Интерфейс должен быть очеловечен.

Революция совершилась с появлением цифровых дисплеев. Теперь можно было взаимодействовать с машиной на уровне "человеческого" диалога. До дисплеев были только циферки и лампочки, а теперь появились буковки. Кстати, буковки были английские (потому что русские еще не были нарисованы/воткнуты в компьютеры). Кстати, устройств, различающих большие и маленькие буквы тоже не было.

Следующая революция произошла, когда появились графические дисплеи. Появилась возможность выводить кривые, прямые и различные образы. Картинки и образы лучше воспринимаются большинством людей. Поэтому блок-схемы понятнее большинству людей, нежели сухой код.

Человек видит только пользовательский интерфейс, но не знает "потрохов" программы, поэтому 99% успешности продукта — пользовательский интерфейс.

Диалоговые системы должны что-то отвечать на запросы пользователя. Лучшими считаются прогресс-бар и извещения о прогрессе выполнения операций.

Методология разработки программного обеспечения.

Главный принцип программирования — "разделяй и властвуй", то есть, структурная декомпозиция.

1) Сначала нужно произвести декомпозицию сложной системы на подсистемы.

2) Оценить сложность каждой подсистемы.

3) Подсчитать суммы $\pm 10\%$ на каждую подсистему, сложить их...

4) И умножить на два!

Умножать на два нужно потому, что интеграционный процесс (процесс сборки системы из подсистем) очень сложен и может занять времени и денег больше, чем написание самих подсистем.

feasibility study — оценка осуществимости.

Итак, к нам пришел заказчик, хочет дать бабла за программу. Мы говорим: давайте мы заключим с вами такой договор: я посажу 3 лучших системных аналитиков, они за 3 месяца решат, можно ли сделать эту программу, после этого мы поговорим подробнее. Стоить это будет столько-то.

Scope — содержимое (объем) работы. За это и заплатит заказчик. Если он согласится с вами работать, то вперед. Если нет — пусть торгуется или идет с этим скоупом к другой компании.

Технология программирования — книжка на сайте Терехова.

Конец шестой лекции.

Quality assurance (QA).

На зачете тем, кому попадется 12 вопрос, следует сказать, что:

1) Мы говорим только о больших программных комплексах (мелкие программки лесом), которые пишутся большими коллективами, и когда мы говорим про методологию, нужно говорить о планировании работы, принятии, обеспечении стандартов и правильной документации. Не бывает единой методологии в программировании. Методология начинается с построения коллектива, затем способ разработки. Прежде всего нужно сделать декомпозицию, при этом модули, на которые вы разделяете программу, должны быть небольшими. Проверить, поддерживают ли используемые технологии подобную декомпозицию. Декомпозиция — основной принцип проектирования (не программирования!)

2) Нужно представить систему в виде образов. Но проблема в том, что не все понимают, что ты имел в виду под тем или иным образом. Это дело решили стандартизировать. И три конкурента: Буч, Рэм-

бо и Якобсон придумали собственные use case. И в 1996 году их троих купила одна компания, заперла на острове и не выпускала, пока они не согласовали собственные разработки в одну. И таким образом получился язык UML. И там стандартизированы способы проектирования. Короче, прогуглим :)

Тестирование и отладка.

Отладка — поиск ошибок (debugging).

Прежде чем запускать программу с тестом, нужно понять, какой ответ выдаст этот тест при исполнении программы.

Тесты гоняют для того, чтобы найти ошибку, а не для того, чтобы проверить правильность выполнения программы. Хороший тест — это тест, на котором программа сломалась. А если тест прошел — то это говно-тест.

QA должен быть адвокатом пользователя и не взаимодействовать с разработчиками.

Структурное программирование — постепенная декомпозиция программы с использованием заранее ограниченного числа конструкций (while, if, разделение на функции и так далее).

Цикл проверять минимум три раза: без захода программы в цикл, с одним заходом, с двумя и так да-

лее.

Лучший способ отладки — чтение программ. Еще лучший способ — описать действия этой программы в статье (и пускай эта статья будет для себя).

Аксиомы программирования:

1) В каждой программе есть ошибка.

2) Если в программе нет ошибок, то ошибка в алгоритме.

3) Если нет ошибок ни в программе, ни в алгоритме, то такая программа никому нафиг не нужна!

При получении задачи нужно думать не о том, как ты напишешь программу, а о том, как ты её будешь тестировать. (test first).

Итоги:

1) Тесты нужно писать до программы!

2) При написании программы нужно использовать принципы структурного программирования (и никаких goto!).

3) Нужно заранее готовиться к отладочной работе! (расставить контрольные точки и понять, что будет делать программа при этом и только после этого нажимать кнопку «Пуск»).

model checking — суть в том, что мы берем программу и по ней строим модель по некоторым предикатам («никогда в будущем не случится...», «обязательно произойдет...», «может произойти при условии...») Это называется темпоральной логикой (TL).

Она отличается от математической тем, что в темпоральной логике присутствует течение времени, которое способно влиять на верность того или иного утверждения.

Среды разработки. Инструменты тестирования и отладки

IDE — интегрированное средство разработки (среда программирования, которая позволяет тебе программировать быстрее).

В чем состоит интегрированность? Мы раньше говорили о простых программах, состоящих из одного модуля. А может быть множество модулей, из которых требуется собрать с помощью makefile единую сущность.

(int a; real b; nest "f1"; char c; ...)

egg "f1" = (b := a; nest "f2")

egg "f2" = (...)

DSL — предметно-ориентированный язык (язык программирования, приближенный к какой-либо предметной области).

Проблема DSL в том, что для каждого DSL требуется создать собственную IDE, собственный генератор кода и т.д.

Метатехнология — это произвольный процесс воздействия на технологию, преобразующий ее от неко-

торого начального к некоторому конечному состоянию.

Пример метатехнологии: QReal, TRIK studio.

ResqueWare

Реижиниринг — переработка старых программ с помощью современных технологий.

Регрессионное тестирование — тестирование старыми тестами (которые уже проходили) программы после внесения изменений (например, после внесения изменений в 50-ю функцию может перестать работать работавшая ранее 20 функция).

CVS (Control Version System) — система "разделения труда" между программистами.

Метрика сложности программы содержит:
1) Количество строк кода;
2) Число функций;
3) Число баз данных и таблиц;
4) Число окошек;
5) Число ветвлений;
6)
Метрика Маккейба (не путать с Макеевым) :)

Доказательство корректности программы.
Метод флойда (возможно).

model checking — суть в том, что мы берем программу и по ней строим модель по некоторым предикатам («никогда в будущем не случится...», «обязательно произойдет...», «может произойти при условии...») Это называется темпоральной логикой (TL). Она отличается от математической тем, что в темпоральной логике присутствует течение времени, которое способно влиять на верность того или иного утверждения.

Дырка в том, что мы проверяем корректность не программы, а модели. А кто поручится за то, что модель соответствует программе? Никто.

model checking архиважен для систем реального времени, но он не панацея, потому что построение модели по программе производится людьми.

Социальный контекст компьютинга и так далее.

Чем отличается профессиональный программист от чайника:

– Он думает о результатах (о будущем своей программы).

– Профессиональный программист работает за деньги :) (то есть его работа кому-то нужна)

– Профессионал соблюдает кодекс этики.

– Профессионал охотно делится своими знаниями с соседями.

– Входит в профессиональные ассоциации (IEEE, ACM, Руссофт)

Авторские права (воровать нехорошо).

IP — интеллектуальная собственность.

Во всех контрактах первое утверждение — что данная программа принадлежит заказчику (не тебе).

Моральные права (ваша фамилия может быть упомянута в программе). Да, это собственность компании, но упомянуть автора они могут (а желательно — чтобы были обязаны).

Конец седьмой лекции.

ИСТОРИЯ ЭВМ И ПРОГРАММИРОВАНИЯ

Стандартная схема ЭВМ (см. фото) появилась в 1836 году (Чарльз Бэббидж)

Процессор	Память	Ввод/Вывод
Шина		
Регистры	Управление	

Аугуста Лавлейнс — первая программистка мира

Первые компьтеры XXI века:

(1916) 1917 — триггер (катодное реле) (М. А. Бонч-Бруевич рулит!)

1939 — ABC, (Д. Атанасов, К. Бэрри)

1941 — Z3 (К. Цузе) (уже двоичная машина). Была целая серия (Z1, Z2, Z3), но Z3 первая, документация на которую была опубликована

1943 — Mark I (Г. Эйкен, Т. Уотсон-младший)

1946 — ENIAC (Д. Моучли, Д. Эккерт, Г. Голдстайн)

Гарвардская архитектура: командная память и память данных физически разделена

Триггер — электронное устройство (схему см. на фото) (по-русски — **катодное реле**) (и ещё раз — изобрёл М. А. Бонч-Бруевич!)

Д. Атанасов изобрёл первый в мире сумматор, но патент был отозван судом США

Джон (Иоганн) фон Нейман — венгерско-американский учёный еврейского происхождения, автор (ну почти автор) современной архитектуры компьютера

Машина фон Неймана: командная память и память данных в одном устройстве

Алан Тьюринг, помимо теоретической деятельности, занимался практическими проектами, в частности, расшифровкой перехваченных сообщений вермахта

Конец XIX века — российский учёный шведского происхождения Однер изобрёл арифмометр и наладил его производство (Терехову, а значит и нам, оно неинтересно)

1. Ламповые ЭВМ

Первые советские ЭВМ:

1951 — МЭСМ, (С.А. Лебедев)

1952 — М-1 (И.С. Брук) (свидетельство от 4.12.1948) (уже полупроводниковая машина)

1952 — БЭСМ (С.А. Лебедев)

1953 — «Стрела» (Ю.Я. Базилевский, Б.И. Рамеев)

1953 — М-2 (М.А. Карцев)

1958 — БЭСМ-2 (В.А. Мельников)

1958 — М-20 (С.А. Лебедев, М.К. Сулим, М.Р. Шура-Бура)

Сергей Александрович Лебедев (1902–1974)

Исаак Семёнович Брук (1902–1974)

Юрий Яковлевич Базилеский (1912–1983)

Башир Искандерович Рамеев (1918–1994)

Михаил Александрович Карцев (1923-1983)

Владимир Александрович Мельников (1928–1993)

Первые советские полупроводниковые ЭВМ:
БЭСМ-3М, БЭСМ-4, М-220, М-220М, М-222
1956 — М-3 (Н.Я. Матюхин)
1963 — Минск-2/22 (В.В. Пржиялковский)
1968 — Минск-32 (В.В. Пржиялковский)
Николай Яковлевич Матюхин (1927–1984)

Оказывается, что наиболее удобная система с точки зрения теории информации — система с основанием e (да-да, число Эйлера)

1959 — «Сетунь» (троичная ЭВМ, единственная в мире) (Н.П. Брусенцов)

Николай Петрович Брусенцов (1925–)

Конец восьмой лекции.

Объектно-ориентированное программирование

Структурное программирование — разбиение задачи на малые подзадачи и решение их по отдельности и композиция их в одну.

Однако повторное использование кода в других проектах (особенности частей кода) затруднено, поскольку «вытащить» её, не затронув другие части

программы (статические переменные, другие функции), очень трудно, почти невозможно.

Реинжиниринг.

Структурное программирование — разбиение проекта на объекты, которые содержат саму по себе отдельную программу и данные, с которой данная программа работает. Подобные объекты взаимодействуют друг с другом и, таким образом, повторное использование подобного объекта становится очень простым.

Первый язык с поддержкой подобной технологии появился в 1967 году.

Что такое объектно-ориентированная парадигма?

Признаки:

1) Инкапсуляция;
2) Наследование;
3) Полиморфизм.

Каждый class — это тип, к которому относятся какие-то объекты. Реализация: смотри class.cpp

Стек - простейший класс. Стек - это некий объект, имеющий 2 интерфейса: push — положить что-либо в стек, pop — взять со стека положенное туда ранее значение.

Конец девятой лекции

Введение в компьютерную графику

Цветовые модели RGB и CMY. Представляют собой задачу цвета по трехмерным координатам относительно выбранных координатными осями цветов. Причем CMY обратна RGB.

Растровая графика — графика, задаваемая по пикселям. Пиксель характеризуется цветом.

DPI — характеристика, показывающая число пикселей на дюйм.

Глубина цвета — сколько возможных цветов мы можем записать посредством такого-то числа битов памяти.

Достоинства растровой графики:

1) Она очень простая (берем и рисуем).

2) Все изображения, какие только могут быть, могут быть сохранены в таком формате.

Недостатки:

1) При масштабировании изображения пиксели становятся видимыми невооруженным глазом.(с ними можно поздороваться :))

2) Такие изображения занимают больно дофига места.

В общем, хороший формат, но не очень.

Векторная графика.

Изображение — совокупность контуров из линий, многоугольников, окружностей, эллипсов, текста и кривых Безье.

Как нарисовать кривую? Очевидно, что она является графиком многочлена. Найдя таковой многочлен, мы сможем хра-

нить его коэффициенты, а не само изображение, таким образом, получается выигрыш в памяти.

Прочитать про кривые Безье.

Достоинства:

1) Размер хранимых изображений;

2) Отсутствие артефактов при зуммировании.

Недостатки:

1) Обычные (не геометрически правильные) изображения очень трудно зарисовать в векторном виде с помощью примитивов.

2) Перевод изображения из растра в вектор охренеть какой непростой.

API — интерфейс прикладного программирования. Интерфейс - это нечто, через что что-то взаимодействует с чем-то.

Есть различные графические библиотеки, которые позволяют взаимодействовать с графической подсистемой операционной системы.

Низкоуровневые операции хороши тем, что работают где угодно, но работать с ними очень неудобно.

Мы будем рассматривать библиотеки, входящие в фреймворк Qt.

Все, что находится в данном фреймворке, абстрагировано, что позволяет работать с любой ОС.

Двумерная графика в Qt реализована с помощью классов:

1) QPainter — эта хрень рисует.

2) QPaintDevice — то, на чем рисует.

3) QPaintEngine — помогает второму рисовать.