01.09.2024

Введение в язык C

Филиппов Михаил Витальевич

m.filippov@g.nsu.ru

89232283872

Императивное программирование, 2024-2025

Новосибирский государственный университет

*НАСТОЯЩАЯ НАУКА

Давайте познакомимся

Филиппов Михаил Витальевич

- Окончил магистратуру ФФ НГУ
- Окончил аспирантуру ИТ СО РАН
- Являюсь м.н.с. ИТ СО РАН
- 7+ лет опыт в программировании C/C++

Адженда

Информация о курсе	Рождение языка С	Зачем нужны алгоритмы	Введение в язык С
25 минут	30 минут	10 минут	25 минут

Адженда

Информация о курсе	Рождение языка С	Зачем нужны алгоритмы	Введение в язык С
25 минут	30 минут	10 минут	25 минут

Правила

1. Иметь подготовленное рабочее пространство
2. Быть пунктуальными
3. Соблюдать сроки работы и договоренности

О предмете

Предмет называется «Императивное программирование». На нем мы будем изучать:

1. Язык Си
2. Алгоритмы и структуры данных

Курс будет читаться в течении двух семестров.

- В первом семестре будет дан набор задач каждую неделю (всего 13 наборов задач). Каждая задача будет решаться в одном файле (пример. main.c). В конце семестра будет дифференцированный зачет.
- Во втором семестре будет дан набор задач раз в две недели (всего 6 наборов задач). Будут рассмотрены: консоль, пути, много файлов, библиотеки, линковка, и т.д. В конце семестра будет экзамен.

О предмете

Типы задач в набое в зависимости от сложности:
базовые/основные/повышенной сложности/космической сложности.

Как сдать задачу:

1. Решить задачу
2. Сдать ее на портале nsuts (когда пройдут все тесты, будет написано **accepted**)
3. Сдать задачу семинаристу

Наказание за списывание

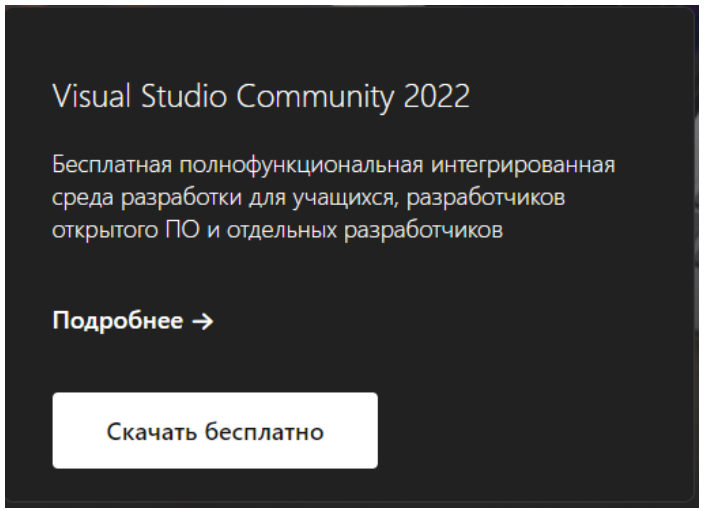
1. симметричное наказание
2. обнуление баллов
3. штрафные очки, если базовые задачи (их надо отработать).


У кого в школе было программирование?

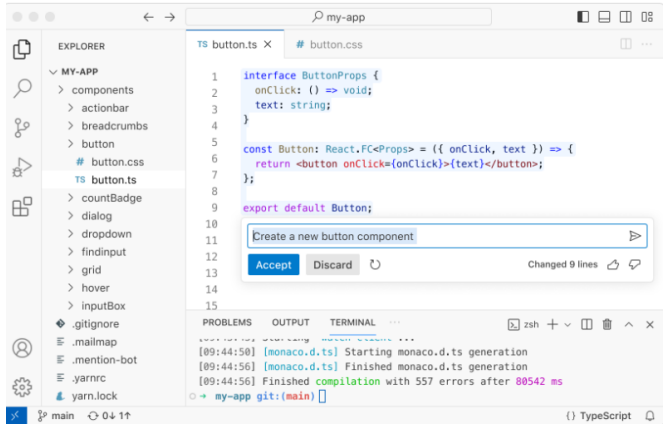
Кто писал на языке Си?

Кто работает на Windows/Linux/MacOs?

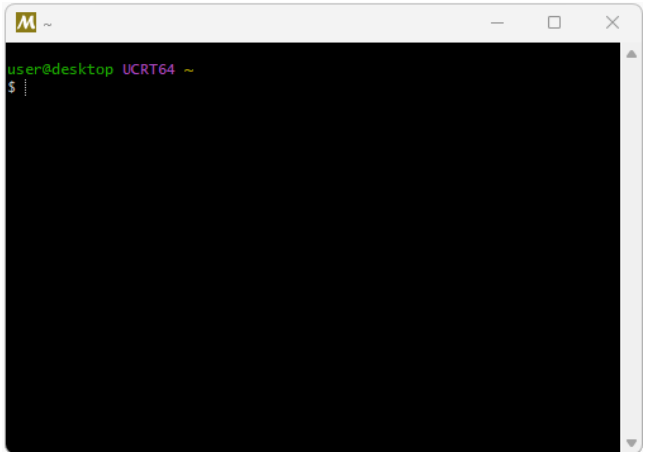
Инфраструктура для учебы



https://visualstudio.microsoft.com/ru/vs/community/



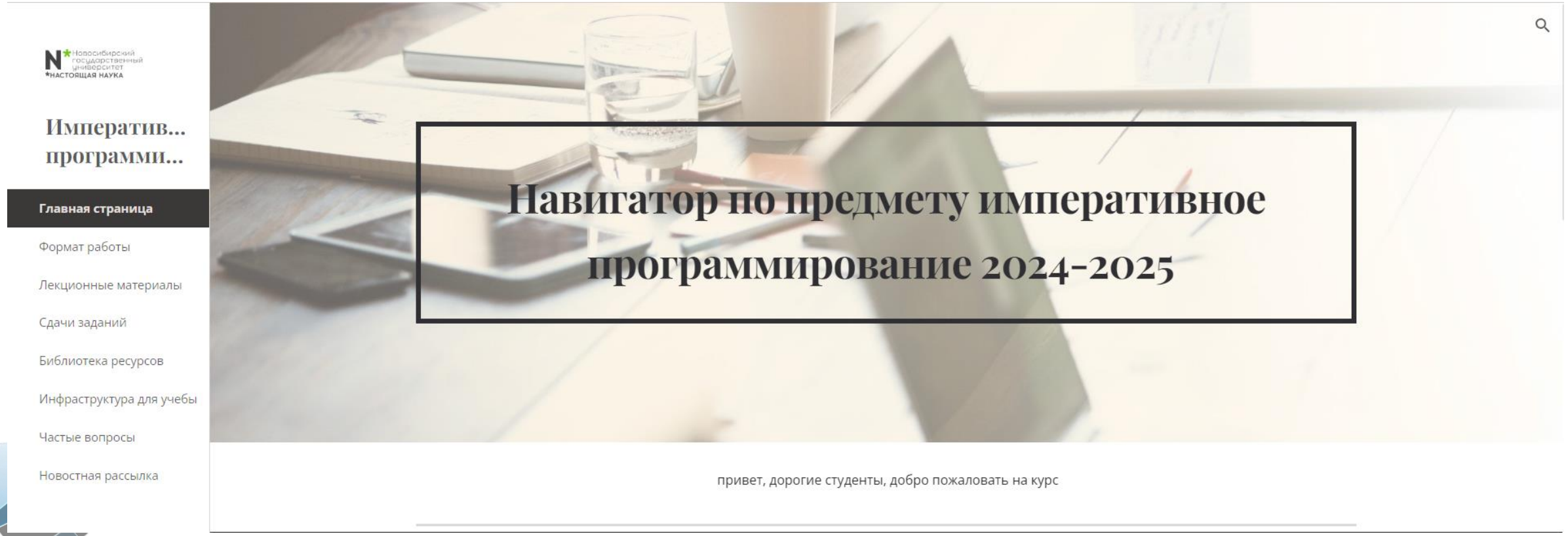
https://code.visualstudio.com/



https://www.msys2.org/

https://sites.google.com/g.nsu.ru/m-filippov/%D0%B8%D0%BD%D1%84%D1%80%D0%B0%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0-%D0%B4%D0%BB%D1%8F-%D1%83%D1%87%D0%B5%D0%B1%D1%8B

САЙТ



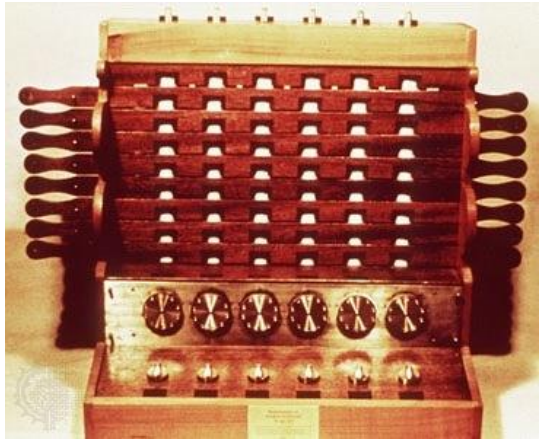
https://sites.google.com/g.nsu.ru/m-filippov/

Адженда

Информация о курсе	Рождение языка С	Зачем нужны алгоритмы	Введение в язык С
25 минут	30 минут	10 минут	25 минут

Вычислительные машины

1 вычислительная машина - Вильгельма Шиккарда 1623 год



автоматически выполняла сложение, вычитание, умножение и деление

https://habr.com/ru/companies/ua-hosting/articles/367083/

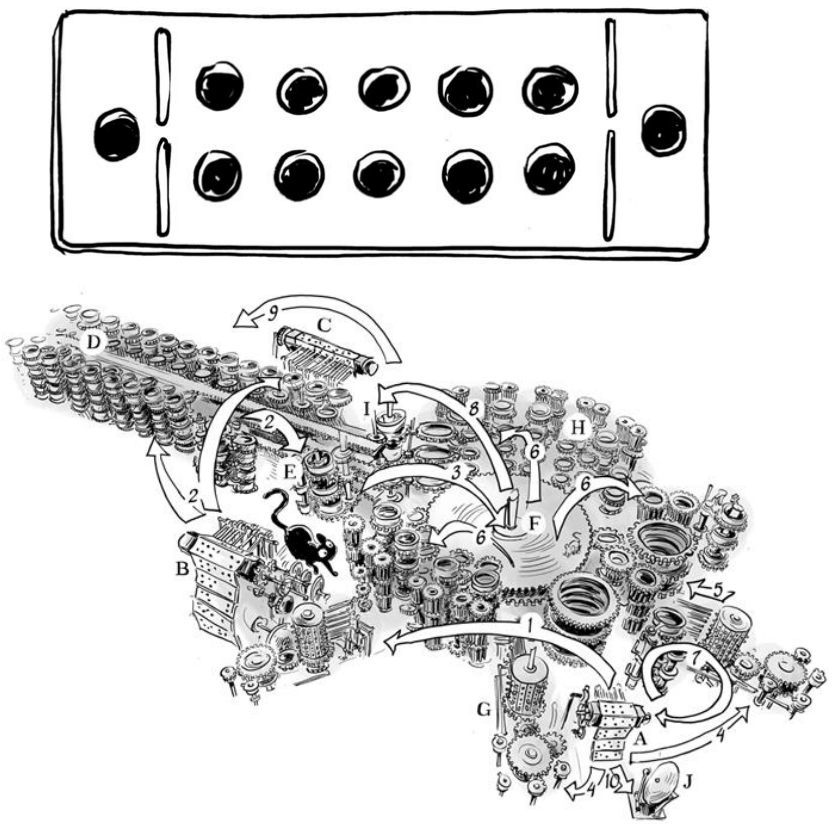
Продвинутую идею в 19 веке предложил Чарльз Беббидж сначала в виде **разностной машины** (1822 год), а позже – **аналитической**

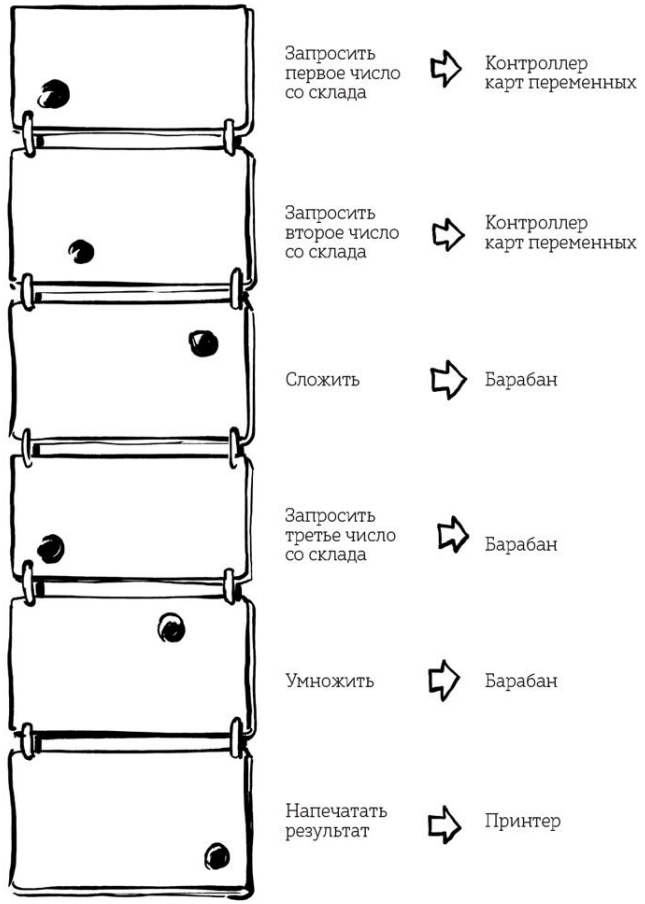
идея программного управления вычислениями

https://habr.com/ru/articles/408223/

Карты

В то время программ не существовало, ну точнее они уже были придуманы, но тогда они назывались картами операций и выглядели примерно так:





Запросить первое число со склада → Контроллер карт переменных

Запросить второе число со склада → Контроллер карт переменных

Сложить → Барабан

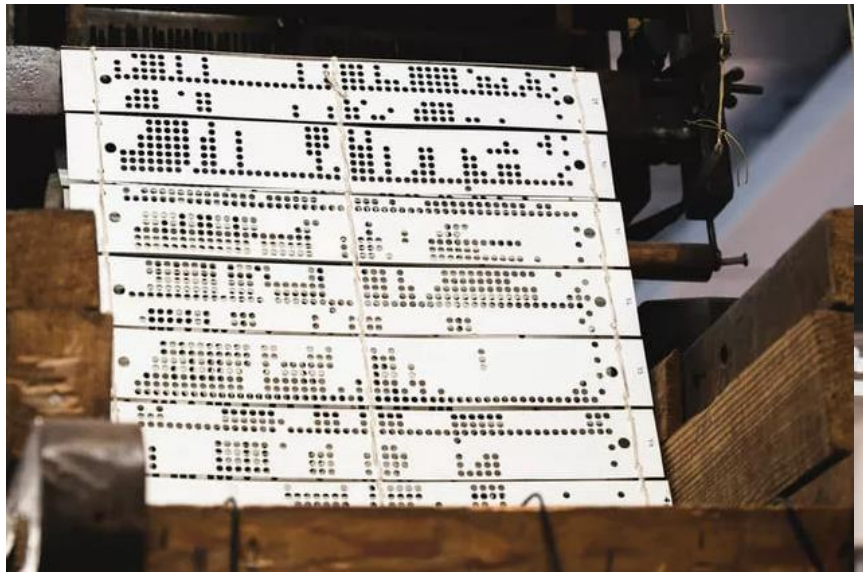
Запросить третье число со склада → Барабан

Умножить → Барабан

Напечатать результат → Принтер

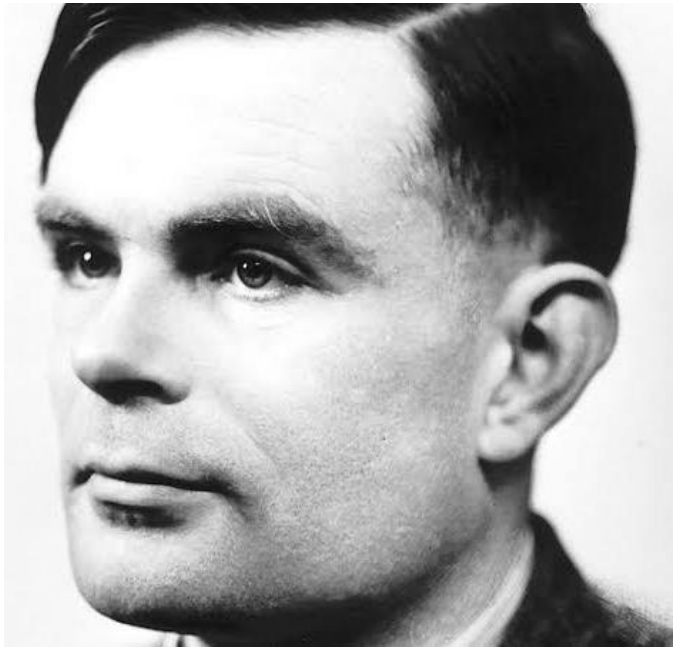
Перфокарты

Первой системой, построенной на перфокартах, был жаккаров станок, и именно им вдохновлялся Бэббидж.



Только в 1980 году перфокарты получили широкое применение. В то время американец Герман Холлерит разработал устройство, которое позволило значительно ускорить процесс подсчёта результатов переписи населения Соединенных Штатов Америки.

Зарождение электроники



Алан Тьюринг
(1912 – 1954 гг.)

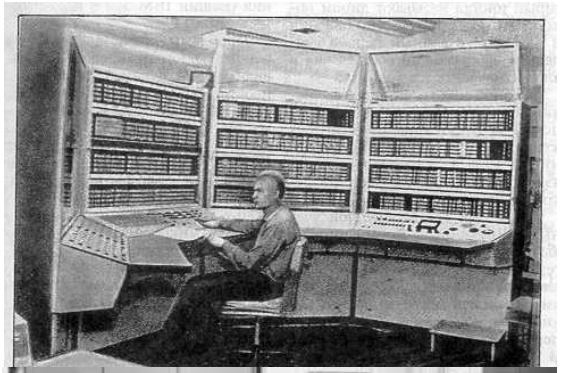
идея универсальной машины Тьюринга

Джон фон Нейман
(1903 – 1957 гг.)

Хранение команды и данных в единой, однородной памяти компьютера

Поколения компьютеров

Первое поколение (1945-1954) - ЭВМ на электронных лампах



ЭВМ второго поколения использовали полупроводниковую элементную базу

ЭВМ третьего поколения использовали в качестве элементной базы интегральные схемы (1968-1973)

Четвертое поколение

ЭВМ четвертого поколения - используют большие и сверхбольшие интегральные схемы (БИС и СБИС), виртуальную память, многопроцессорный с параллельным выполнением операций принцип построения, развитые средства диалога (середина 80-х гг. по настоящее время).

Операционная система

При отсутствии ОС при решении задачи пришлось бы вначале прописывать на программном уровне взаимодействие процессора с дисплеем, клавиатурой, жестким диском. Затем, порядок запуска и выполнения программы вычисления матрицы, контролировать цикл обработки команд процессора, так как он работает непрерывно, и решать еще очень и очень много других вспомогательных задач, не относящихся непосредственно к поставленной перед нами начальной цели. **ОС решает эту задачу.**

ОС также предоставляет набор команд для управления состоянием компьютера для оператора – человека

Как появился С

- зарождение Unix способствовало появлению первой версии языка Си в 1960-70х
- Assemler, BCPL оказались неудобными для создания ОС
- С был создан для ухода с низкого ассемблерного уровня и позволить писать программы на более высоком абстрактном уровне, при этом, сохраняя все преимущества низкоуровневого программирования

```
@loop1:;[
    mov rax, r11;
    fld QWORD PTR [rax];=sub; 1~| sub |
    fld st(0);=sum;           2~| sub | sum |
    add rax, 8;
    fld _2pin;                3~| sub | sum | 2pi/n |

    fild k;                   4~| sub | sum | 2pi/n | k |
    fmulp st(1),st;=w;        3~| sub | sum | w |
    fsincos;                  4~| sub | sum | sin(w) | cos(w) |

    fld1;=u;                  5~| sub | sum | sin(w) | cos(w) | u |
    fldz;=v;                  6~| sub | sum | sin(w) | cos(w) | u | v |
```

Assemler

```
35 S $ = ""
40 FOR I = 1 TO S
50 S $ = S $ + "*"
55 NEXT I
60 PRINT S $
70 INPUT "Do you want more stars?"; Q $
80 IF LEN (Q $) = 0 THEN GOTO 70
90 L $ = LEFT $ (Q $, 1)
100 IF 30 (L $ = "Y") OR (L $ = "y") THEN GOTO
110 PRINT "Goodbye";
120 FOR I = 1 TO 200
130 PRINT A $; "";
140 NEXT I
150 PRINT
```

BCPL

```
#include "stdafx.h"
#include "conio.h"
#include "iostream"

int main() {
    setlocale(LC_ALL, "Rus");

    char word[20];

    printf("Введите строку: ");
    scanf_s("%19s", word, (unsigned)_countof(word));

    int length = 0;
    while (word[length] != '\0') { length++;  }

    printf("Длина строки '%s' равна: %d", word, length);

    _getch();
    return 0;
}
```

C

Преимущества и недостатки языка С

- ✓ Конструктивные особенности
- ✓ Эффективность
- ✓ Переносимость
- ✓ Мощь и гибкость
- ✓ Ориентация на программистов

- ✗ Излишняя ответственность
- ✗ Постоянная бдительность
- ✗ Сложность понимания

Стандарты языка С

- **Первый стандарт ANSI/ ISO C** - доверие программисту, простота и краткость языка, единственность выполнения действия, быстродействие по возможности, переносимость.
- **Стандарт C99** - интернационализация, исправление дефектов и повышение вычислительной полезности
- **Стандарт C11** - смягчение цели “доверия программисту”, послабления по функциональности, добавление необязательной поддержки параллельного программирования и т. д.

ANSI (American National Standards Institute — Национальный институт стандартизации США)
ISO (International Organization for Standardization — Международная организация по стандартизации)

Использование языка С

Комментирование!

Механика программирования

Компиляторы:

- MSVC
- GCC
- Clang

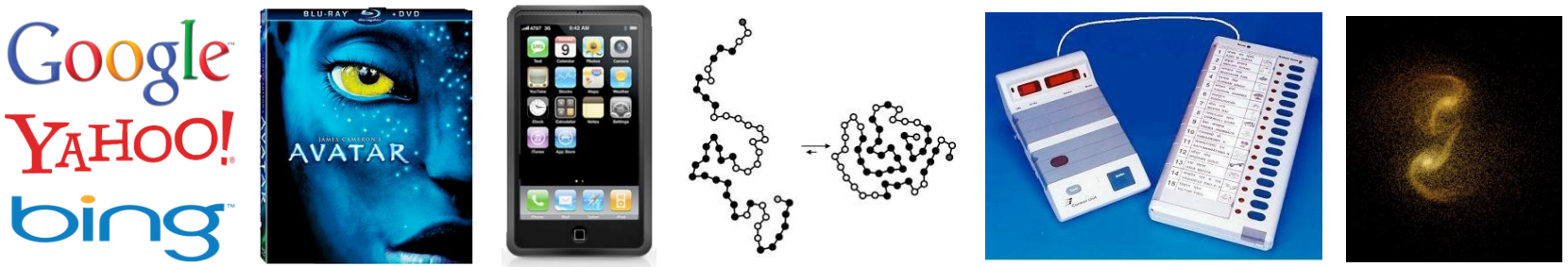
Адженда

Информация о курсе	Рождение языка C	Зачем нужны алгоритмы	Введение в язык C
25 минут	30 минут	10 минут	25 минут

Почему алгоритмы?

Их воздействие широко и далеко идущее.

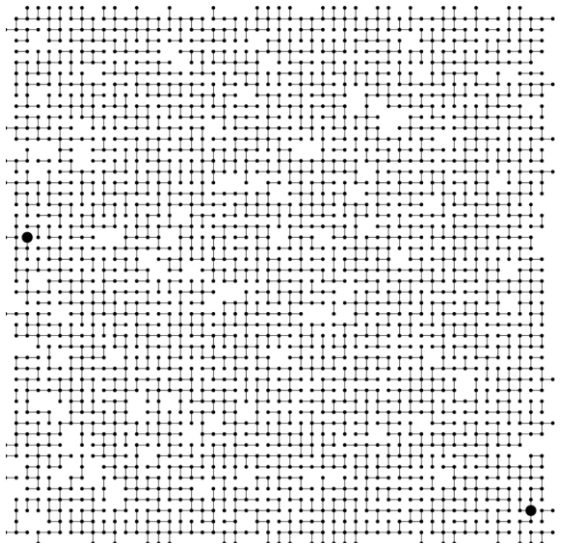
- Интернет. Веб-поиск, маршрутизация пакетов, распределенный обмен файлами...
- Биология. Проект генома человека, сворачивание белка,...
- Компьютеры. Схема схемы, файловая система, компиляторы,...
- Компьютерная графика. Фильмы, видеоигры, виртуальная реальность...
- Безопасность. Сотовые телефоны, электронная коммерция, машины для голосования...
- Мультимедиа. MP3, JPG, DivX, HDTV, распознавание лиц...
- Социальные сети. Рекомендации, новостные ленты, реклама, ...
- Физика. Моделирование N-тел, моделирование столкновений частиц, ...
- …

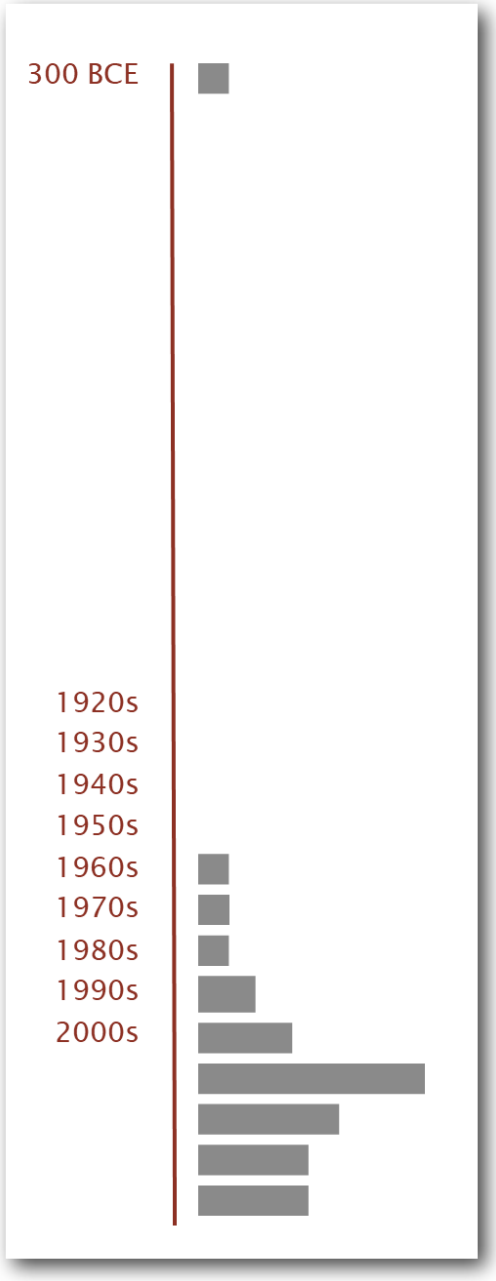


Почему алгоритмы?

Старые корни, новые возможности.

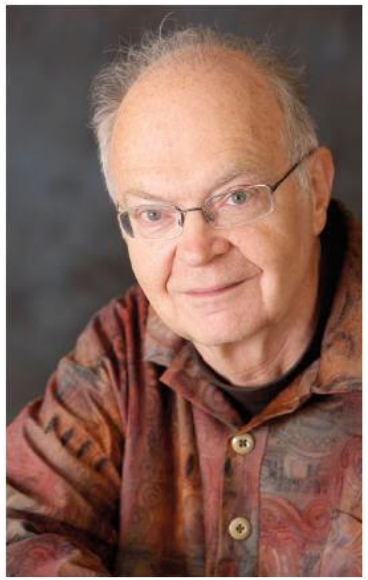
- Изучение алгоритмов восходит от Евклида.
- Концепция алгоритмов формализована Черчем и Тьюрингом в 1930-х годах.
- Были обнаружены некоторые важные алгоритмы от студентов на похожих курсах
- Нужны для решения проблем, которые иначе нельзя решить.





Цитаты классиков

- *“ For me, great algorithms are the poetry of computation. Just like verse, they can be terse, allusive, dense, and even mysterious. But once unlocked, they cast a brilliant new light on some aspect of computing. ” — Francis Sullivan*
- *“ An algorithm must be seen to be believed. ” — Donald Knuth*
- *“ I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships. ” — Linus Torvalds (creator of Linux)*
- *Algorithms: a common language for nature, human, and computer. ” — Avi Wigderson*



Адженда

Информация о курсе	Рождение языка С	Зачем нужны алгоритмы	Введение в язык С
25 минут	30 минут	10 минут	25 мин

Первая программа на С

```c
#include <stdio.h> /* инструкции препроцессора */

int main(void) /* простая программа */
{
    int num; /* определить переменную с именем num */
    num = 1; /* присвоить значение переменной num */
    printf("Я студент "); /* использовать функцию printf() */
    printf("ИИР!\n");
    printf("Моей любимой цифрой является %d, так как она первая.
\n", num);
    return 0;
}

// Я простой компьютер.
// Моей любимой цифрой является 1, так как она первая.
```

Директивы # include и заголовочные файлы

- Директива **include** (включить файлы) представляет собой удобный способ совместного использования информации.
- Оператор **#include** представляет собой пример директивы **препроцессора** в C. В общем случае компиляторы языка C выполняют некоторую подготовительную работу над исходным кодом перед компиляцией; это называется предварительной обработкой.
- Разработчики языка C называют совокупность информации, которая помещается в верхней части файла заголовком
- Файл **stdio.h (standard input/output header – заголовочный файл стандартного ввода-вывода)** поставляется как часть всех пакетов компиляторов C. Он содержит информацию о функциях ввода и вывода, таких как printf(), и предназначен для использования компилятором

Функция main()

- Программа на языке С (с некоторыми исключениями) всегда начинается с выполнения функции **main()**.
- скобки идентифицируют main() как функцию
- int main (void) - возвращаемый тип функции main() определен как **int**
- void main() – ***не рекомендуется*** (компиляторы не обязаны принимать эту форму)

Комментарии

```
/* Это комментарий на C. */
/* Этот комментарий, будучи несколько многословным,
размещен в двух строках. */
/*
Допустим также и такой комментарий.
*/
```

В стандарте C99 появился еще один стиль комментария, который был популяризирован языками C++ и Java. Новый стиль предполагает применение символов / / для представления комментария, ограниченного одной строкой:

```
// Данный комментарий умещается в одной строке.
int rigue; // Комментарий можно также поместить сюда.
```

Скобки, тела и блоки

{
}

- В общем случае вес функции языка С используют фигурные скобки для обозначения начала и конца своего тела. Для этой цели допускается применять только фигурные скобки { }, но не круглые ( ) или квадратные [ ]
- Фигурные скобки можно также использовать внутри функции для организации операторов в модуль или блок.

Объявления

int num; - Эта строка программы называется оператором объявления

1. где-то в функции имеется переменная по имени num
2. с помощью int переменная num объявлена как целочисленная, т.е. число без десятичной точки, или без дробной части

Объявление соединяет конкретный идентификатор с конкретной ячейкой в памяти компьютера и при этом устанавливает тип информации, или тип данных, которые будут там храниться.

В языке С все переменные должны быть объявлены до того, как они будут использоваться.

Выбор имени

- Для переменных следует выбирать осмысленные имена (или идентификаторы), например, sheep_count вместо x3, если программа занимается подсчетом овец.
- Стандарты C99 и C11 разрешают использовать имена идентификаторов любой желаемой длины, но компилятор должен рассматривать в качестве значащих только первые 63 символа. Более старые компиляторы часто останавливались на максимум 8 символах. Можно использовать больше символов, но компилятор просто не обязан обращать внимание на дополнительные символы.
- В вашем распоряжении имеются буквы нижнего и верхнего регистров, цифры и знак подчеркивания (_).

Допустимые имена	Недопустимые имена
wiggles	$Z] **
cat2	2cat
Hot_Tub	Hot-Tub
taxRate	tax rate
kcab	don' t

Присваивание

num = 1; В рассматриваемом примере это означает “присвоить значение 1 переменной num”.
Можно переопределять
num = 2;
num = 1000;

Функция printf()

printf("Я простой") ;
printf ("компьютер. \n");
printf ("'Моей любимой цифрой является %d, так как она первая. \n", num);

- внутри круглых скобок — это фактические аргументы функции
- printf () просматривает все, что заключено в двойные кавычки, и выводит этот текст на экран.
- Символы \n означают начало новой строки.
- Символ % уведомляет программу, что в этом месте будет выведено значение переменной, а d указывает на то, что переменная должна выводиться как десятичное целое число.

Оператор возврата

return 0;

- Оператор возврата является завершающим оператором программы.
- Функции C, возвращающие значения, делают это с помощью оператора возврата, состоящего из ключевого слова return, за которым следует возвращаемое значение и точка с запятой.

Д/З

1. Установить ПО
2. Разобраться, что такое scanf()
3. Разобраться с операторами +, -, *, /, %
4. Решить пак под номером 0 и показать решение семинаристу

Резюме

- Разобрались с основными формальными аспектами работы
- Посмотрели вкратце историю С
- Немного поговорили о важности алгоритмов и курса
- Разобрали первую программу на Си