Введение в программирование Лекция 1

Лопатин Александр

- 1 О курсе
- 2 Программировать просто?
- 3 Вводные термины
- 4 Обзор развития вычислительных систем
- Парадигмы программирования
- 6 Алгоритмическое программирование
- 7 Структурное программирование

Введение в программирование _О курсе

О чем курс

- урезанная версия одноименного универовского курса (обычно рассчитанного на 1—2 семестра)
- ▶ основы и упоминания из других курсов

Введение в программирование СО курсе

Expectations

- получим кругозор в программировании
- узнаем основы нескольких языков
- научимся базовым техникам
- на практике применим некоторые техники

Как построен курс

- разбит так, чтобы не вызвать передоз информацией
- одно занятие в неделю на 2—3 часа (включает лекцию и практику)
- ▶ 10 недель ($\approx 2\frac{1}{3}$ месяца)
- домашки на не более 2-х часов в неделю
- один мелкий проект на 2 недели (2 викенда)
- домашки и проект будут оцениваться

Введение в программирование — Программировать просто?

Программировать просто?

Программировать просто?

Не знаю

Введение в программирование —Программировать просто?

Зависит от

- Специализации
- Интеллекта
- Способов обучения

Проблема образования

- курсы и книжки рассчитаны на студентов универов
- предполагается что студент имеет интеллект выше среднего
- большинство людей не такие \Rightarrow программирование сложно для среднего человека (на самом деле нет)



Программировать просто?

Корректный вопрос: как ты стал (хорошим) программистом?

Введение в программирование Вводные термины

Вводные термины

Исполнитель

- повар
- военнослужащий
- гитарист
- коммерческая компания
- ▶ вычислительная система (компьютер, смартфон, бытовой прибор и т.п.)

Инструкции исполнителя

- пункты рецепта («42. Положить ложку сахара») для повара
- ▶ приказы («нале-во!») для военнослужащего
- аккорды для гитариста
- инструкции процессора («сложить два числа») для вычислительной системы

Алгоритм

Конечная последовательность **инструкций исполнителя** направленная на решение задачи

- рецепт для повара
- лабораторная работа для студента
- текст песни для вокалиста
- бизнес-план коммерческой компании
- шаги воспроизведения проблемы для тестировщика
- **компьютерная программа** для вычислительной системы

С точки зрения данных

Ввод o Обработка o Вывод

Интерфейс

Означает «взаимодействие» Нужно для осуществления ввода/вывода данных

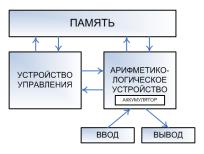
- ▶ Пользовательский (UI User Interface)
 - Графический пользовательский интерфейс, Graphical User Interface (GUI)
 - Интерфейс коммандной строчки, Command-Line Interface (CLI)
- ▶ Программный (API Application Interface)
 - подпрограммы, модули, библиотеки
 - сетевые протоколы (например HTTP)
 - много других

CLI может использоваться как API

Введение в программирование Собзор развития вычислительных систем

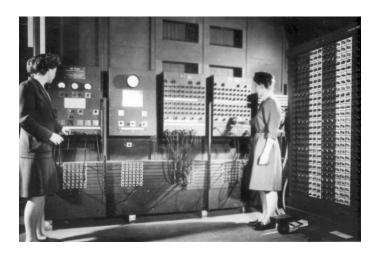
Обзор развития вычислительных систем

1945: Архитектура фон Неймана (Von Neumann architecture)



□Обзор развития вычислительных систем

1946: ENIAC



1964: IBM System/360

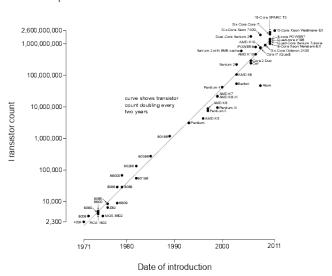


В 60-е значение слова «Хакер» еще не было испорчено журналистами

Стивен Леви — Хакеры: Герои компьютерной революции

1975: Закон Мура

Microprocessor Transistor Counts 1971-2011 & Moore's Law



- выполнять трудоемкие операции на других устройствах (например GPU)
- объединять несколько ядер процессоров в один
- объединять несколько компьютеров в вычислительный кластер
- проектировать концептуально другие выч. системы, необязательно на основе фон Неймовской архитектуры
 - квантовые компьютеры
 - клеточные компьютеры
 - **.**..



С ростом производительности железа растет и сложность задач

Сложность программ тоже растет

Появляется много специализаций в ІТ, по аналогии с врачами

Многоуровневые системы

Системы состоят из подсистем (из слоёв / уровней)

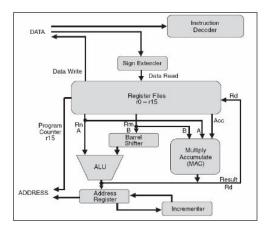


Уровни (levels / layers) в реальной жизни Уровни характеризуются обязанностями и определяют кто над кем главнее (выше)

- Директор конторы (высокий уровень)
 - Художник (средний уровень)
 - Программист (средний уровень)
 - Младший программист (низкий уровень)

Обзор развития вычислительных систем

Каждая железяка состоит из множества более низкоуровневых железок



Софт тоже многоуровневый

Your program How users see it user interface user interface logic MAGIC data

Введение в программирование — Парадигмы программирования

Парадигмы программирования

Введен	ние в программирование			
∟Пар	адигмы программирования			
			Декларативн	pe I
			1	1
			1	I I
			I .	1
			I I	I I
			i	i
			1	
			Ī	i
	I Императивное		1	
	I :	1	i	i
	l I		1	I I
	I	1	i	i
	 	1	1	1
	I	i	i i	i
	 		1	I
	i I	i	i i	i I
	 	1	1	!
	i I	i		
	1	1		
	' 	i		
	1	1		
	! 			
	!	1		
	1 			
	1	1		

Императивное программирование (Imperative Programming)

Написание **алгоритмов** путём перечисления **инструкций исполнителя**



Декларативное программирование (Declarative Programming)

Описание желаемого результата, вместо алгоритма получения этого результата.

Например запрос к базе данных:

Выбрать имена абитуриентов поступающих на специальность «Программная Инженерия» с сортировкой по сумме баллов по убыванию

Подробнее рассмотрим ближе к концу курса

Введение в программирование			
Алгоритмическое программирование			
		Декларативное	1
		i i	i
		!	
		1	
		İ	1
		i.	i
		!	
	-1	! 	
императивное	1	I.	
	i	1	
!	1	i .	i
	- 1	1	
Алгоритмическое	1	i	i
	-	1	
Блок-схемы, словесное описание	i	1	i
1	- 1	I .	1
!	i	L	
Į.	1		
	-		
i	i		
1	- 1		
	i		
!	1		
	1		
	- 1		

Введение в программирование — Алгоритмическое программирование — Словесное описание

Линейный алгоритм

- 1. Шагнуть вперед
- 2. Шагнуть вперед
- 3. Повернуть направо на 90°
- 4. Шагнуть вперед

Алгоритм с ветвлением

- 1. Если небо светлое перейти к п. 2 иначе к п. 4
- 2. Напечатать «Сейчас день»
- Перейти к п.5
- 4. Напечатать «Сейчас ночь»
- 5. Напечатать «Ваш Капитан Очевидность»

Введение в программирование — Алгоритмическое программирование — Язык блок-схем (flow-charts)

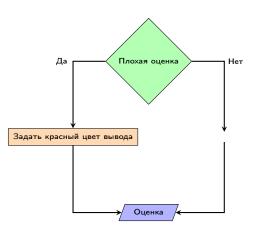
Линейный алгоритм



Введение в программирование — Алгоритмическое программирование — Язык блок-схем (flow-charts)

Алгоритм с ветвлением

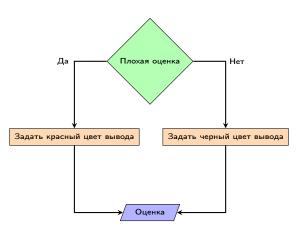
Бывает с одной веткой



Введение в программирование — Алгоритмическое программирование — Язык блок-схем (flow-charts)

Алгоритм с ветвлением

И с двумя ветками

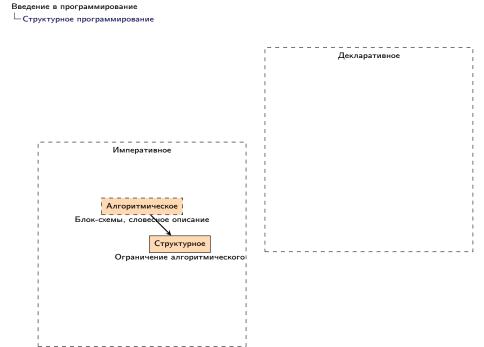


Введение в программирование
— Алгоритмическое программирование
— Язык блок-схем (flow-charts)

Практика



- 1. Нарисовать блок-схему к программе, которая берет в качестве ввода числа a и b и решает уравнение $a=b\cdot c$. Вывести решение (значение c). (Подсказка: сначала нужно вручную преобразовать уравнение)
- 2. Предусмотреть обработку ошибки деления на ноль к предыдущему заданию. Выводить «Ошибка: деление на ноль» вместо решения в этом случае
- 3. Написать словесное описание получившегося алгоритма



Структурное программирование (Structured Programming)

- отказ от меток и безусловного перехода
- использование двух управляющих структур (control structures)
 - условие
 - ▶ цикл

Spaghetti Sample Structured Sample 1.058 JMPL RBW SWITCHING 1.058 JMPL ResolutionBandwidth 1.059 LABEL FREQUENCY READOUT 1.059 LABEL ResolutionBandwidth done 1.060 CALL Sub Frequency Readout 1.060 JMPL ResolutionSwitching 1.061 CALL Sub Frequency Span 1.062 CALL Sub Noise Sidebands 1.061 LABEL ResolutionSwitching_done 1.063 JMPL SYSTEM SIDEBANDS -1.062 JMPL DisplayedAverage 1.064 LABEL WIDE OFFSETS 1.063 LABEL DisplayedAverage done 1.065 CALL Sub Noise Wide Offsets 1.064 JMPL ResidualResponses 1.066 JMPL SPURIOUS 1.065 LABEL ResidualResponses done 1.067 LABEL SYSTEM SIDEBANDS 1.066 JMPL AbsoluteAmplitude 1.068 CALL Sub System Sidebands 1.069 CALL Sub Residual 1.067 LABEL AbsoluteAmplitude done 1.070 CALL Sub Display Switching 1.068 JMPL FrequencyReadout 1.071 JMPL SCALE FIDELITY 1.069 LABEL FrequencyReadout done 1.072 LABEL SWEEP_TIME -1.070 JMPL FrequencySpan 1.073 CALL Sub Sweep Time 1.071 LABEL FrequencySpan done 1.074 JMPL WIDE OFFSETS 1.072 JMPL NoiseSidebands 1.075 LABEL SCALE FIDELITY 1.073 LABEL NoiseSidebands done 1.076 CALL Sub Scale Fidelity 1.077 CALL Sub Input Switching 1.074 JMPL SystemSidebands 1.075 LABEL SystemSidebands done 1.078 CALL Sub Reference Level 1.079 JMPL FREQUENCY RESPONSE 1.076 JMPL Residual 1.080 LABEL RBW SWITCHING 1.077 LABEL Residual done 1.081 CALL Sub Resolution BW Switching 1.078 JMPL Display Switching 1.082 JMPL RESOLUTION BANDWIDTH . 1.079 LABEL Display Switching done 1.083 LABELABSOLUTE AMPLITUDE -1.080 JMPL ScaleFidelity 1.084 CALL Sub Absolute Amplitude 1.085 JMPL FREQUENCY READOUT. 1.081 LABEL ScaleFidelity_done 1.086 LABEL RESOLUTION_BANDWIDTH -1.082 JMPL InputSwitching 1.087 CALL Sub Resolution Bandwidth 1.083 LABEL InputSwitching done 1.088 JMPLAVERAGE NOISE 1.084 JMPL ReferenceLevel 1.089 LABEL FREQUENCY RESPONSE 1.085 LABEL ReferenceLevel done 1.090 CALL Sub Freg Response 1.086 JMPL FregResponse 1.091 JMPL SWEEP TIME 1.087 LABEL FreqResponse done 1.092 LABEL AVERAGE NOISE 1.093 CALL Sub Displayed Average 1.088 JMPL SweepTime 1.094 JMPL RESIDUAL RESPONSES = 1.089 LABEL SweepTime done 1.095 LABEL SPURIOUS -1.090 JMPL NoiseWideOffsets 1.096 CALL Sub TOI 1.091 LABEL NoiseWideOffsets done 1.097 JMPL GAIN COMPRESSION -1.092 JMPL TOI 1.098 LABEL 2ND_HARMONIC -1.093 LABEL TOI done 1.099 CALL Sub 2nd Harmonic 1.094 JMPL GC 1.100 CALL Sub Other Spurs 1.101 JMPL END 1.095 LABEL GC done 1.102 LABEL GAIN COMPRESSION 1.096 JMPL 2ndHarmonic 1.103 CALL Sub GC

1.104 JMPL 2ND HARMONIC

1.105 LABEL RESIDUAL RESPONSES 4

1.106 CALL Sub Residual Responses 1.107 JMPLABSOLUTE AMPLITUDE: 1,108 LABEL END -

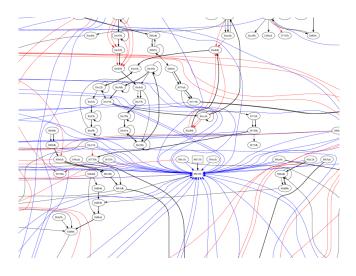
1.097 LABEL 2ndHarmonic

1.098 JMPL OtherSpurs

1.099 LABEL OtherSpurs

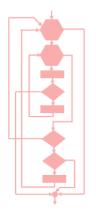
Структурное программирование

Spaghetti is write-only code



Повторения всё же нужны

Но не такие безобразные



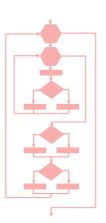
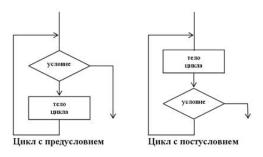


Рис.: Спагетти-код и структурный код

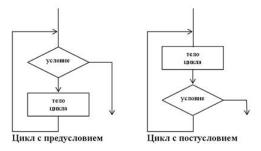
Структурное программирование

Циклы



Цикл с предусловием (while)

- 1. Пока гвоздь не забит
 - 1.1 Поднять молоток
 - 1.2 Ударить молотком по гвоздю
 - 1.3 Перейти к п. 1



Цикл с постусловием (do ... while)

- 1. (Делать)
 - 1.1 Поднять молоток
 - 1.2 Ударить молотком по гвоздю
- 2. Пока гвоздь не забит перейти к п. 1



Цикл со счетчиком (for) — частный случай цикла с предусловием

- 1. Инициализировать счетчик і значением 1
- 2. Пока i <= n
 - 2.1 Выполнить действие
 - 2.2 Выполнить другое действие
 - 2.3 ...
 - 2.4 Увеличить счетчик і на единицу
 - 2.5 Перейти к п. 2.

Всё тот же цикл for

- 1. Для \underline{i} от $\underline{1}$ до \underline{n} с шагом $\underline{1}$
 - 1.1 Выполнить действие
 - 1.2 Выполнить другое действие
 - 1.3 ...
 - 1.4 Перейти к п. 1.

Циклы могут быть вложены

- 1. Для <u>i</u> от <u>1</u> до <u>количество_студентов</u> с шагом <u>1</u>
- 2. Напечатать имя і-го студента
 - 2.1~Для $\underline{\mathrm{j}}$ от $\underline{\mathrm{1}}$ до $\underline{\mathrm{количество}}$ оценок $\underline{\mathrm{студентa}}$ с шагом $\underline{\mathrm{1}}$
 - 2.1.1 Показать ј-ю оценку і-го студента
 - 2.1.2 Перейти к п. 2.1
 - 2.2 Перейти к п. 1
- 3. Напечатать «такие дела»

Практика

- 1. Нарисовать блок-схему к алгоритму забивания гвоздя со слайда с циклом с предусловием
- 2. Тоже самое для слайда с циклом с постусловием
- 3. Тоже самое для <u>алгоритма с вложенными циклами</u> (подсказка: можно начать с вложенного цикла)

Домашка

Нарисовать <u>блок-схему</u> и написать <u>словесное описание</u> к алгоритму решения квадратного уравнения $ax^2+bx+c=0$. Входные данные: $a,\ b$ и c. Выходные данные:

- либо «Корней нет»
- либо x₁
- либо x₁ и x₂

Подсказки:

- lacktriangle дискриминант вычисляется по формуле $D=b^2-4ac$
- lacktriangle корни уравнения вычисляются по формуле $x_{1,2}=rac{-b\pm\sqrt{D}}{2a}$
- количество корней зависит от значений дискриминанта (3 ситуации: D < 0, D = 0 и D > 0)