Объектно-ориентированное программирование
Лекция 1. Общее введение

П. А. Макаров

СЫКТЫВКАРСКИЙ
ЛЕСНОЙ
ИНСТИТУТ

16 сентября 2022 г.

Содержание

fg=Green

Знакомство и основные сведения

- Дисциплина — Объектно-ориентированное программирование;
- Преподаватель: Макаров Павел Андреевич, makarovpa@ipm.komisc.ru;
- Трудоёмкость — 16 ч. лекций + 16 ч. лаб. раб;
- Форма итоговой аттестации — экзамен;
- Сайты в поддержку курса:
  - https://www.makarovpa.ru/oop;
  - https://mp.komisc.ru:1987.

fg=Green

Литература

Основная

1. *Вайсфельд М.* Объектно-ориентированное мышление.
2. *Мейер Б.* Объектно-ориентированное конструирование программных систем.
3. *Пышкин Е. В.* Основные концепции и механизмы объектно-ориентированного программирования.
4. *Пол А.* Объектно-ориентированное програмирование на C++.
5. *Буч Г.* Объектно-ориентированный анализ и проектирование с примерами приложений на C++.
6. *Андрианова А. А., Исмагилов Л. Н., Мухтарова Т. М.* Объектно-ориентированное програмирование на C++.
7. *Страуструп Б.* Язык программирования C++.
8. *Липпман С., Лажойе Ж.* C++ для начинающих.
9. *Столяров А. В.* Введение в язык Си++.
10. *Подбельский В. В.* Язык Си++.
11. `http://www.cplusplus.com`.
12. *Пилгрим М.* Погружение в Python 3.
13. `https://www.python.org/doc/`.
14. `https://en.wikibooks.org/wiki/Python_Programming`.

fg=Green

Литература

Дополнительная

1. *Кнут Д.* Искусство программирования.
2. *Вирт Н.* Алгоритмы и структуры данных.
3. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы. Построение и анализ.
4. *Реймонд Э. С.* Искусство программирования для Unix.
5. *Брайант Р., О'Халларон Д.* Компьютерные системы: архитектура и программирование.
6. *Столяров А. В.* Программирование: введение в профессию.
7. *Керниган Б., Ритчи Д.* Язык программирования C.
8. *Подбельский В. В., Фомин С. С.* Программирование на языке Си.
9. *Хэзфилд Р., Кирби Л.* Искусство программирования на C: Фундаментальные алгоритмы, структуры данных и примеры приложений.
10. *Stallman R.* GNU C Language Intro and Reference Manual.

fg=Green

Что такое программа?

Определение 1

Программа *— это текст, представляющий собой исчерпывающее описание действий исполнителя и написанный на понятном ему языке.*

Определение 2

Компьютер *— любое электронное устройство независимо от его конкретной архитектуры, способное исполнять программы.*

Определение 3

Язык программирования *— это формальный язык, предназначенный для записи компьютерных программ. Язык программирования представляет собой совокупность лексических, синтаксических и семантических правил.*

fg=Green

- По уровню применяемых абстракций.
- По режиму исполнения программ.
- По используемой типизации.
- По поддержке различных компьютерных архитектур.
- По назначению.
- По возможности следования парадигме программирования.

```cpp
#include <iostream>

using namespace std;

int main() {
    cout << "Hello, world!" << endl;
    return 0;
}
```

Листинг 1: Пример текста, написанного на языке C++

```python
#!/usr/bin/python3.8

print("Hello, world!")
```

Листинг 2: Пример текста, написанного на языке Python

fg=Green

Определение 4

Исходный код *— это текст компьютерной программы на каком-либо языке программирования или языке разметки, который может быть прочтён человеком. Исходный код транслируется в исполняемый код целиком до запуска программы при помощи транслятора либо исполняется непосредственно при помощи интерпретатора. В широком смысле, исходный код — это любые входные данные для транслятора.*

Определение 5

Транслятор *— это программа, выполняющая* ***трансляцию****, то есть преобразование программы, представленной на одном из языков программирования, в программу на другом языке.*

Процесс трансляции программ, написанных на языке C++

1. Обработка исходного текста *препроцессором*.
2. *Компиляция*.
3. *Ассемблирование*.
4. *Компоновка*.

Определение 6

Интерпретатор *— программа, выполняющая* ***интерпретацию*** *— построчный анализ, обработку и выполнение исходного кода программы.*

Виды интерпретаторов

1. Простые.
2. Компилирующего типа.

Алгоритм работы простого интерпретатора

1. прочитать инструкцию;
2. проанализировать её и определить требуемые действия;
3. выполнить необходимые действия;
4. если не достигнуто условие завершения программы, прочитать следующую инструкцию и перейти к пункту 2.

Некоторые интерпретаторы могут работать в режиме диалога или *цикла чтения-вычисления-печати* (REPL — read-eval-print loop).

Пример 1

Фреймворк ROOT, используемый для анализа данных в области физики высоких энергий, предоставляет пользователю возможность работать в режиме диалога с помощью интерпретаторов языков C++, Python либо R.

- ISO/IEC 14882:1998 (C++98).
- ISO/IEC 14882:2003 (C++03)
  `https://cs.nyu.edu/courses/fall11/CSCI-GA.2110-003/documents/c++2003std.pdf`.
- ISO/IEC 14882:2011 (C++11).
- ISO/IEC 14882:2014 (C++14).
- ISO/IEC 14882:2017 (C++17).
- ISO/IEC 14882:2020 (C++20).

Основные документы, “стандартизирующие” Python

- The Python Language Reference.
- The Python Standard Library.
- PEP 0 — Index of Python Enhancement Proposals.
- PEP 8 — Style Guide for Python Code.
- PEP 257 — Docstring Conventions.

Определение 7

Архитектура компьютера *— это концептуальная модель компьютерной системы, определяемая её компонентами и их взаимодействиями между собой и окружением. Кроме того, в понятие архитекуры включаются принципы её проектирования и развития.*

Уровни организации компьютерной архитектуры

-1. Уровень физических устройств.
0. Цифровой логический уровень.
1. Микроархитектурный уровень.
2. Уровень архитектуры системы команд.
3. Уровень операционной системы.
4. Уровень языка ассемблера.
5. Язык высокого уровня.

Определение 8

Операционная система *— это комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.*

Основные этапы развития компьютеров и операционных систем

- Пакетный режим.
- Разделение времени и многозадачность.
- Разделение полномочий.
- Системы реального времени.
- Гибридные системы.

Нужны ли операционные системы?

Операционные системы используются

- если нужен универсальный механизм хранения данных;
- для предоставления программам системных библиотек с часто используемыми подпрограммами;
- для распределения полномочий;
- необходима возможность имитации “одновременного” исполнения нескольких программ на одном компьютере;
- для управления процессами выполнения отдельных программ.

Современные универсальные ОС

- используют файловые системы;
- многопользовательские;
- многозадачные.

fg=Green

- Основные функции:
  - загрузка программ в оперативную память и их выполнение.
  - исполнение запросов программ.
  - стандартизованный доступ к периферийным устройствам.
  - управление оперативной памятью.
  - управление доступом к данным на носителях, организованным в определённой файловой системе.
  - обеспечение пользовательского интерфейса.
  - сохранение информации об ошибках системы.
- Дополнительные функции:
  - параллельное или псевдопараллельное выполнение задач.
  - эффективное распределение ресурсов вычислительной системы между процессами.
  - разграничение доступа различных процессов к ресурсам.
  - организация надёжных вычислений.
  - взаимодействие между процессами: обмен данными, взаимная синхронизация.
  - защита системы и данных/программ пользователя.
  - многопользовательский режим и разграничение прав.

Группы компонентов ОС

- Ядро и планировщик. Драйверы устройств, непосредственно управляющие оборудованием. Сетевая и файловая системы.
- Системные библиотеки.
- Оболочка с утилитами.

Определение 9

Ядро *— центральная часть операционной системы, управляющая ресурсами компьютера, выполнением процессов и обеспечивающая координированный доступ процессов к ресурсам.*

fg=Green

Основные ресурсы системы

- процессорное время;
- память;
- устройства ввода-вывода;
- файловая система;
- сетевая система.

Основные объекты ядра

- Процессы.
- Файлы.
- События.
- Потоки.
- Семафоры.
- Мьютексы.
- Каналы.

fg=Green

Определение 10

Системный вызов *— это элемент пользовательского интерфейса ядра, представляющий собой обращение прикладной программы к ядру операционной системы для выполнения той или иной операции.*

Определение 11

Программный интерфейс приложения (API — Application Programming Interface) *— это описание способов взаимодействия одной компьютерной программы с другими.*

Определение 12

Двоичный интерфейс приложений (ABI — Application Binary Interface) *— это набор соглашений для доступа приложения к операционной системе и другим низкоуровневым сервисам, спроектированный для переносимости исполняемого кода между машинами, имеющими совместимые ABI.*

Основы компьютерных сетей

- Клиент-серверная архитектура.
- Сетевая модель OSI.
- Стек протоколов TCP/IP.
- Интерфейс.
- Порт.
- Сокет.

fg=Green

Инструменты разработчика

Какие инструменты нужны программисту?

Инструменты разработчика

- Текстовый редактор.
- Набор транслятора либо интерпретатор.
- Библиотека необходимых функций и компонентов.
- Система сборки.
- Отладчик.
- Профилировщик кода.
- Система контроля версий.
- Генератор документации.

Определение 13

Интегрированная среда разработки *— это комплекс программных средств, используемый программистами для разработки программного обеспечения (IDE — Integrated Development Environment).*

fg=Green

Инструменты разработчика

- Текстовый редактор. `vim`
- Набор транслятора либо интерпретатор. `gcc/python3.8`
- Библиотека необходимых функций и компонентов. `glib`
- Система сборки. `make`
- Отладчик. `gdb`
- Профилировщик кода.
- Система контроля версий. `git`
- Генератор документации.

Определение 13

Интегрированная среда разработки *— это комплекс программных средств, используемый программистами для разработки программного обеспечения (IDE — Integrated Development Environment).* *`Code::Blocks`*

fg=Green

Контрольные вопросы и задания

1. Найдите и изучите информацию о HTML, CSS, Perl, PHP, Java, JavaScript, SQL, Bash, TEX, Asymptote, Gnuplot. Является ли всё перечисленное языками программирования? Почему? Классифицируйте из данного списка всё, что является языками программирования.
2. Что такое язык разметки? Язык сценариев? Компьютерный язык? Как соотносятся все эти понятия с термином язык программирования?
3. Что такое полнота по Тьюрингу и какое это имеет отношение к программированию?
4. Найдите информацию о наиболее популярных трансляторах языка C++ и изучите их основные особенности. Существуют ли интерпретаторы языка C++?
5. Какие интерпретаторы Python вам известны? К какому типу они относятся и каков принцип их работы?
6. С какими архитектурами компьютеров вы работали? Каковы их основные особенности? Наидите и изучите информацию о неизвестных вам архитектурах.
7. Назовите существующие интерфейсы взаимодействия пользователя с операционной системой. Каковы их преимущества и недостатки?
8. Определите значения понятий, перечисленных в разделе 7.
9. Каковы, на ваш взгляд, преимущества и недостатки использования IDE для разработки ПО? С какими инструментами разработчика у вас есть опыт работы?

fg=Green