Программа учебной дисциплины: Язык прогаммирования

Утверждена
Кафедрой компьютерной безопасности
МИЭМ НИУ ВШЭ
Протокол № 5 от «24» июня 2019 г.

Академическим советом ОП 25.06.2019 г.

Автор	Лебедев П.А. (plebedev@hse.ru)
Число кредитов	12
Контактная работа (час.)	192
Самостоятельная работа (час.)	264
Курс	1 курс – 2 курс 1 семестр
Формат изучения дисциплины	Без использования онлайн курса

I. ЦЕЛЬ, РЕЗУЛЬТАТЫ ОСВОЕНИЯ ДИСЦИПЛИНЫ И ПРЕРЕКВИЗИТЫ

Основной целью освоения дисциплины «Языки программирования» является формирование базовых компетенций, связанных с разработкой программного обеспечения при решении профессиональных задач. В рамках дисциплины вырабатываются навыки программирования и алгоритмизации с применением современных процедурных и объектно-ориентированных языков программирования. Параллельно с рассмотрением данных языков, демонстрируется их связь с языком ассемблера. За счёт рассмотрения основных представителей языков различного уровня и их связей, закладывается понимание иерархии существующего ПО, происходит освоение разнообразного инструментария и подготовка специалиста к проектированию и анализу программных систем.

В результате освоения дисциплины студент должен:

- знать
  - общие принципы построения и использования современных языков программирования высокого уровня;
  - язык программирования высокого уровня (объектно-ориентированное программирование);
  - особенности взаимодействия языков высокого и низкого уровня, организации работы с памятью в скриптовых языках.
  - базовые структуры данных;
  - основные комбинаторные и теоретико-графовые алгоритмы, а также способы их эффективной реализации и оценки сложности;
  - современные технологии программирования
- уметь
  - формализовать поставленную задачу;
  - работать с интегрированными средами разработки программного обеспечения;

- разрабатывать системное и прикладное программное обеспечение для многозадачных, многопользовательских и многопроцессорных сред, а также для сред с интерфейсом, управляемым сообщениями;
- проводить оценку сложности алгоритмов;
- разрабатывать эффективные алгоритмы и программы;
- планировать разработку сложного программного обеспечения;
- оценивать качество готового программного обеспечения;

- иметь навыки
  - разработки, документирования, тестирования и отладки программ;
  - навыками использования инструментальных средств отладки и дизассемблирования программного кода;
  - навыками разработки алгоритмов решения типовых профессиональных задач.

Настоящая дисциплина относится к профессиональному циклу дисциплин и блоку дисциплин, обеспечивающих базовую подготовку для всех специализаций.

Изучение данной дисциплины базируется на общешкольных знаниях.

Для освоения учебной дисциплины, студенты должны владеть следующими знаниями и компетенциями:

- Владение математическим аппаратом в рамках программы средней школы;
- Умение использования персонального компьютера для ввода и поиска информации, знание основных возможностей ОС по работе с файлами, умение устанавливать и изучать дополнительное ПО;
- Основы английского языка.

Основные положения дисциплины должны быть использованы в дальнейшем при изучении следующих дисциплин:

- Математическое программное обеспечение;
- Методы программирования;
- Операционные системы;
- Языки ассемблера
- Системно-ориентированное программирование;
- Системы управления базами данных;
- Программирование алгоритмов защиты информации;
- Параллельные вычисления;
- Анализ программных реализаций алгоритмов защиты.

Знания и практические навыки, полученные в результате освоения дисциплины «Языки программирования», используются студентами при разработке курсовых и дипломных работ, в научно-исследовательской работе.

II. СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Раздел 1. Основы процедурного программирования на языке C++

Тема 1. Введение в языки программирования

Тема 2. Использование виртуальных машин. Основы языка оболочки командной строки. Взаимодействие программ со средой выполнения.

Тема 3. Обзор и основные понятия языка C++. Лексический состав языка.

Обзор структуры программы.

Тема 4. Основные арифметические типы данных. Основные операции с арифметическими типами.

Тема 5. Объекты и доступ к ним. Составной оператор. Дополнительные операции записи

Тема 6. Пример компиляции выражения. Функции. Аппаратный стек и автоматическое время хранения.

Тема 7. Условный оператор if. Введение в форматированный ввод-вывод. Простые циклы.

Тема 8. Тернарная операция, оператор for, операторы перехода, константные выражения, оператор switch. Изменение порядка выполнения команд в машинном коде. Стили оформления программ. clang-format.

Тема 9. Пространства имён. Квалифицированные имена. Связанность описаний функций, прототипы и рекурсия. Анонимные пространства имён. Статическое время хранения. Описание и директива using, псевдонимы областей видимости.

Тема 10. Использование компилятора из командной строки. Undefined Behavior Sanitizer. Среда Qt Creator и система сборки CMake. Средства статического анализа: clang-tidy.

Тема 11. Инициализация копированием и её использование в определениях и операторах ветвления и цикла. Псевдонимы типов. Типы фиксированной ширины. Перегрузка функций и аргументы по умолчанию. Математические функции стандартной библиотеки.

Тема 12. Возможности предварительной обработки. Заголовочные файлы. Программы из нескольких единиц трансляции.

Тема 13. Указатели на отдельные объекты. Операция sizeof. Простое использование массивов. Комбинирование конструкций создания производных типов. Указатели на функции.

Тема 14. Квалификаторы типов, const-корректность. Расширенные константные выражения. Нулевой указатель. Средства динамического анализа: Address Sanitizer, Valgrind.

Тема 15. Побитовые операции.

Тема 16. Строковые литералы. Функции классификации символов и работы с нуль-терминированными строками. Передача информации об ошибках между функциями.

Раздел 2. Основные структуры данных. Архитектура программ и проектов.

Тема 1. Сложность вычислений. Работа с динамической памятью. Функции работы с последовательностями тривиальных типов.

Тема 2. Графы. Многомерные структуры данных. Введение в классовые типы.

Тема 3. Графовые структуры данных: связанные списки, деревья. Представления в виде массивов: куча.

Тема 4. Хэш-таблицы. Сравнение динамических структур данных.

Тема 5. Встраиваемые функции. Неполные типы и инкапсуляция в стиле C. Утверждения и тесты.

Тема 6. Библиотеки.

Тема 7. Системы контроля версий: git.

Раздел 3. Основы объектно-ориентированного программирования на языке C++.

Тема 1. Прочие виды инициализации. Временные объекты. Ссылки.

Тема 2. Классовые типы и их члены: статические и нестатические члены данных и функции члены, псевдонимы типов, вложенные классы. Локальные классы.

Тема 3. Инициализация классов: конструкторы. Пользовательские преобразования типов. Перегрузка операций. Аргументо-зависимый поиск имён. Друзья классов.

Тема 4. Специальные функции-члены класса.

Тема 5. Перечисления. std::byte. Intrinsic-функции и конфигурационные проверки в системах сборки, has_include.

Тема 6. Наследование классов.

Тема 7. Виртуальные функции и виртуальное наследование. RTTI. typeid и dynamic_cast.

Раздел 4. Обобщённое программирование на языке C++.

Тема 1. Концепции. Шаблоны функций и классов. Дедукция для шаблонов функций и классов. Инстанциация и специализация шаблонов. std::enable_if.

Тема 2. Простые применения шаблонов. Комплексные числа. std::array/std::span/std::string_view. Вариадические шаблоны. Свёртки.

Тема 3. Стандартная библиотека шаблонов. Простые концепции. Итераторы. Контейнеры. Последовательности. Универсальный доступ к контейнерам.

Тема 4. Строки и виды в них. Регулярные выражения. Адаптеры контейнеров. std::bitset.

Тема 5. Цикл for для диапазона. Функциональные объекты. Лямбда-выражения. Алгоритмы стандартной библиотеки.

Тема 6. Детали реализации контейнеров. Работа с памятью для нетривиальных типов, раздельная с инициализацией/уничтожением. Перегрузки операций new/delete. Аллокаторы. Работа с неинициализированной памятью. Диспетчеризация по тегам. if constexpr.

Тема 7. Пары, кортежи. Ассоциативные контейнеры. Структурные привязки.

Раздел 5. Дополнительные возможности C++

Тема 1. Smart-указатели. std::optional/std::any. Объединения и std::variant.

Тема 2. Подходы к обработке ошибок: std::exception/std::error_code.

Тема 3. Потоки ввода/вывода.

Тема 4. Дата/время. (Псевдо)случайные числа и распределения.

Тема 5. Язык C: отличия.

Тема 6. Использование библиотек языка C в C++.

Тема 7. Обзор библиотек boost.

Тема 8. Сборка и использование сторонних библиотек.

Тема 9. Работа с файловой системой.

Тема 10. Многопоточное программирование в модели с общей памятью. Потоки. Упорядоченность операций. std::thread. Простые примеры без общего доступа к объектам.

Тема 11. Примитивы синхронизации: мьютексы и условия. Атомарные

операции.

Раздел 6. Разработка программ с графическим интерфейсом пользователя.

Тема 1. Библиотеки Qt. QObject и метакомпилятор. Контейнеры и QString.

Тема 2. QVariant. QIODevice, QDataStream/QTextStream. clazy.

Тема 3. QWidget. Обзор Виджетов. QPainter. QDialog / QInputDialog / QMessageBox.

Тема 4. Дизайнер форм и UIC, Layouts. Ресурсы и RCC.

Тема 5. Виджеты с архитектурой модель/вид для элементов и графических объектов.

Тема 6. QRunnable/QThread и использование фоновых операций.

Тема 7. Интернационализация.

Тема 8. Обзор прочих модулей библиотек Qt.

Тема 9. Паттерны программирования.

Тема 10. Приёмы документирования исходного кода.

III. ОЦЕНИВАНИЕ

Формы контроля:

Тип контроля	Форма контроля	1 курс				2 курс		Примечания
		1 м	2 м	3 м	4 м	1 м	2 м	
Текущий	Домашняя работа	*	*	*	*	*	*	
	Контрольная работа			*			*	
	Курсовая работа						*	
Промежуточный	Экзамен в устной форме			*				
Итоговый	Экзамен в устной форме						*	

Оценки по всем формам текущего контроля выставляются по 10-балльной шкале.

Основной формой текущего контроля являются задачи, подлежащие теоретической защите. Допускаются дополнительные задачи, не требующие защиты.

При защите выполненных домашних заданий помимо самой практической работы отдельно учитывается теоретическая подготовка студента по соответствующей теме, соответствующая самостоятельной работе. Теоретические знания и практические умения студента оцениваются отдельно, оценкой за работу студента по конкретному заданию является меньшая из оценок за теорию и практику. Общая оценка за домашние задания и самостоятельную работу $O_{д/з}$ является взвешенной суммой оценок за выполненные в отчётный период задачи, веса которых назначаются преподавателем пропорционально их объёму и сложности. В неё может быть включена оценка активности на практических занятиях $O_{ауд}$ по усмотрению преподавателя, их проводящих.

Накопленная оценка $O_{накопленная}$ за текущий контроль вычисляется по формуле:

$$O_{накопленная} = 0{,}8 \cdot O_{д/з} + 0{,}2 \cdot O_{к/р}$$

Промежуточная оценка за дисциплину рассчитывается следующим образом:

$$O_{промежуточная} = 0,6 \cdot O_{накопленная\ этапа\ 1} + 0,4 \cdot O_{промежуточный\ 'экзамен}$$

Накопленная оценка по итогам обучения вычисляется по формуле:

$$O_{накопленная\ итоговая} = 0,5 \cdot O_{промежуточная} + 0.5 \cdot O_{накопленная\ этапа\ 2}$$

В диплом выставляет результирующая оценка по учебной дисциплине, которая формируется по следующей формуле:

$$O_{результ} = 0,6 \cdot O_{накопленная\ итоговая} + 0,4 \cdot O_{итоговый\ экзамен}$$

Во всех формулах вычисления оценок округление осуществляется в сторону ближайшего целого, ровно одна вторая в дробной части округляется вверх.

Контрольная работа используется для оценки способности учащегося работать с программным кодом, автором которого он не является. Выдаваемый учащемуся текст программы заведомо может содержать различные ошибки и недочёты. Условием выставления отличной оценки является нахождение всех синтаксических и семантических ошибок и наличие минимум одной корректной и значительной рекомендации по улучшению предъявленной реализации, при корректном указании всех характеристик сущностей, присутствующих в примере. Допускается вариант с теоретическими вопросами по имеющемуся тексту с выставлением оценки пропорционально числу верных ответов.

Зачёт и экзамен могут включают в себя элементы, аналогичные контрольной работе, общетеоретические вопросы и практические вопросы по последнему домашнему заданию или итоговому проекту.

IV. ПРИМЕРЫ ОЦЕНОЧНЫХ СРЕДСТВ

Примеры итоговых проектов

- Реализуйте построение планарной формы графа, заданного пользователем в виде ввода матрицы инцидентности в таблицу в графическом интерфейсе программы. Раскладку графа выполнить с помощью библиотеки graphviz, преобразовав набор кривых, выдава-емых в результате раскладки, в команды интерфейса векторной графики библиотеки Qt без использования промежуточных файлов.
- Разработайте программу, практически оценивающую, являются ли те или иные форматы файлов сжатыми. Типы файлов определяются расширением. Критерием того, что файл является сжатым, является малое изменение его размера при повторном сжатии. Для тестового сжатия файла используйте одну из сторонних библиотек. Программа сканирует все файлы в заданном каталоге и собирает статистику по сжимаемости всех встреченных типов файлов. Результаты представьте в виде текстового отчёта с предпо-ложениями о характеристиках типов файлов.
- Реализуйте визуализатор сетевого траффика. Задача программы – в реальном времени отображать графическое представление количества информации, передаваемое и принимаемое данным компьютером по сети с разбиением по протоколам. Например, две обновляемые в реальном времени круговые диаграммы (на приём и передачу),

каждый сектор которой – удельный объём относительно всего сетевого траффика, используе-мый протоколом или группой протоколов (WWW/Почта/пиринговые сети/служебный траффик/другое…) за последние 3 секунды. Возможно использование любых визуаль-ных представлений, использующих графику, при сохранении наглядности. Для захвата сетевого траффика использовать библиотеку libpcap.

V. РЕСУРСЫ

1. Основная литература

- Лебедев П.А. C++: устройство и применение (рабочая версия). (Доступно для учащихся по дисциплине в материалах LMS)

2. Дополнительная литература

- Stroustrup B. Programming: Principles and Practice Using C++ (2nd Edition). Addison-Wesley Professional; 2014 (Перевод на русский: Страуструп Б. Программирование: принципы и практика с использованием C++. М. Вильямс, 2017, второе издание).
- Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ. М. Вильямс, 2017

3. Программное обеспечение

№	Наименование	Условия доступа
1.	Образ виртуальной машины с ОС Linux и предустановленным ПО для выполнения заданий по дисциплине	Создание копий для учащихся по дисциплине

4. Профессиональные базы данных, информационные справочные системы, интернет-ресурсы (электронные образовательные ресурсы)

№	Наименование	Условия доступа
1.	Вузовская электронно-библиотечная систе-ма учебной литературы http://miem.hse.ru/	URL: http://miem.hse.ru/
2.	База научно-технической информации (ВИНИТИ РАН)	*Из внутренней сети университета (договор)*
3.	C & C++ reference	URL: https://cppreference.com
4.	Intel® 64 and IA-32 Architectures Software Developer Manuals	URL: https://software.intel.com/en-us/articles/intel-sdm
5.	Qt Documentation	URL: http://doc.qt.io/

5. Материально-техническое обеспечение дисциплины

Для проведения занятий по дисциплине необходимо презентационное оборудование (ПК, проектор, экран).

Для проведения практических занятий необходимо наличие компьютерного класса, оборудованного ПК из расчёта одно рабочее место на обучаемого. На данных ПК требуется наличие соответствующих прав доступа для установки ПО, перечисленного в п. 3. Для упрощения работы с материалами желательно наличие доступа к внешним носителям. Для упрощения доступа к справочным материалам и заданиям, подготовленным с использованием LMS и других сетевых технологий, очень желательно наличие доступа к сети Интернет.