Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
Московский государственный университет имени М.В. Ломоносова
филиал МГУ в г. Севастополе
факультет компьютерной математики
кафедра программирования

УТВЕРЖДАЮ
Директор
Филиала МГУ в г. Севастополе
_____ О.А. Шпырко
«15» июня 2020 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)
Наименование дисциплины (модуля):

СИСТЕМЫ ПРОГРАММИРОВАНИЯ

код и наименование дисциплины (модуля)

Уровень высшего образования:
бакалавриат

Направление подготовки:

01.03.02 Прикладная математика и информатика

(код и название направления/специальности)

Направленность (профиль) ОПОП:
общий
(если дисциплина (модуль) относится к вариативной части программы)

Форма обучения

очная

Рабочая программа рассмотрена
на заседании кафедры программирования
протокол № 3 от «28» апреля 2020 г.
Руководитель ОП 01.03.02 «Прикладная математика и информатика»
_____(Н. В. Лактионова)
(подпись)

Рабочая программа одобрена
Методическим советом
Филиала МГУ в г.Севастополе
Протокол № 6 от «10» июня 2020 г.
_____(А.В. Мартынкин)
(подпись)

Севастополь, 2020

Рабочая программа дисциплины (модуля) разработана в соответствии с самостоятельно установленным МГУ образовательным стандартом (ОС МГУ) для реализуемых основных профессиональных образовательных программ высшего образования по направлению подготовки 01.03.02 «Прикладная математика и информатика» в редакции приказа МГУ от 30 декабря 2016 г.

Год (годы) приема на обучение 2016,2017 ,2018

курс – II
семестр – 4
зачетных единиц – 3 кредита
академических часов – 108, в т.ч.:
лекции – 54 часа
практические занятия – нет
самостоятельная работа – 54 часа
Формы промежуточной аттестации:
зачёты в семестрах – нет
Форма итоговой аттестации:
зачет с оценкой в 4 семестре.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

1. Цель и задачи освоения дисциплины

Целью изучения дисциплины «Системы программирования» является формирование у студентов научного подхода к организации процесса разработки сложных программных продуктов на базе использования теории формальных языков, грамматик и принципов объектно-ориентированного программирования (ООП).

- Изучаются основные принципы организации объектно-ориентированной технологии разработки индустриальных программных продуктов. Рассматриваются и анализируются возможности современных систем программирования.
- Обучение основано на использовании объектно-ориентированной парадигмы программирования, реализованной средствами языка программирования Си++ в системах разработки поддерживающих объектно-ориентированное программирование.
- Рассматриваются элементы теории формальных языков, грамматик и их применение для построения основных элементов систем программирования синтаксических анализаторов и трансляторов. Реализация принципов ООП рассматривается на примере разработки интерпретатора модельного языка.

2. Место дисциплины в структуре образовательной программы

Дисциплина входит в блок профессиональных дисциплин вариативной части образовательной программы. Логически и содержательно изучаемый материал связан с курсами «Алгоритмы и алгоритмические языки», «Архитектура ЭВМ и язык ассемблера» и «Операционные системы».

Курс поддерживается дисциплиной Учебная практика (Практикум на ЭВМ) входящей в блок Практики ОС МГУ по направлению подготовки 01.03.02 «Прикладная математика и информатика».

3. Требования к результатам обучения по дисциплине

В результате освоения дисциплины обучающийся должен:

Знать:

- основные принципы организации систем программирования и процесса разработки сложных программных продуктов
- основные принципы, положенные в основу ООП (абстракция, инкапсуляция, наследование и полиморфизм)
- принципы, положенные в основу синтаксического анализа и синтеза формальных языков;

Уметь:

- анализировать возможности современных систем программирования и производить выбор системы программирования нужной для эффективного решения прикладных задач;

- применять принципы ООП для решения прикладных задач;
- использовать средства формальных языков и грамматик для описания процессов решения прикладных задач.

Владеть:

- профессиональными знаниями в области организации и использования современных систем программирования
- навыками решения практических задач теории трансляции, задач объектно-ориентированного программирования;
- методами теории трансляции, объектно-ориентированного программирования;

Универсальные, профессиональные и специализированные компетенции, которыми должен обладать студент в результате освоения дисциплины

- применять принципы ООП для описания задач обработки информации (ИК-3);
- использовать средства систем программирования для разработки индустриальных прикладных программ (ПК-3).

4. Структура и содержание дисциплины

а) Общая трудоёмкость дисциплины

- 3 зачетных единицы,
- 108 академических часов. В том числе: 54 часа лекций, 54 часа самостоятельной работы

б) Тематический план

Таблица 1.

№ п/п	Название темы	Количество часов			Формы текущего контроля успеваемости (по темам). Форма промежуточной аттестации (по семестрам)
		Л	**С(П,Лб)**	**СРС**	
1	**2**	**3**	**4**	**5**	**6**
Раздел 1. Введение					
1	Этапы жизненного цикла программного продукта. Технологии разработки программного обеспечения. Компоненты систем программирования, поддерживающие разработку программного обеспечения	2		2	Консультации
Раздел 2. Принципы ООП					
2	Основные принципы ООП: абстракция, инкапсуляция, наследование, полиморфизм.	2		2	Консультации
3	Обзор средств языка См++, поддерживающих основные механизмы	2		2	Консультации
4	Виды отношений между классами. Взаимодействие и иерархия классов. ER-диаграммы	2		2	Консультации
5	Концепция абстрактных типов данных и ее реализация в Си++. Объектно-ориентированный анализ предметной области. Отображение сущностей предметной области в классы	2		2	Консультации
6	Классы и объекты. Инициализация, конструкторы, деструкторы. Работа с состоянием объекта	2		2	Консультации
1	**2**	**3**	**4**	**5**	**6**

7	Указатели и ссылки на объекты, передача параметров по ссылке. Инициализация, конструкторы, деструкторы	2		2	Консультации
8	Работа с динамической памятью. Статические и константные члены классов	2		2	Консультации
9	Виды полиморфизма в Си++. Статический полиморфизм. Перекрытие имен. Перегрузка функций. Перегрузка унарных и бинарных операций	4		2	Консультации
10	Одиночное и множественное наследование. Правила наследования. Видимость при наследовании. Иерархия классов. Виртуальные функции. Абстрактные классы. Чистые виртуальные функции. Фабрики объектов	4		2	Консультации
11	Средства обработки ошибок. Исключения в Си++. Динамическая идентификация типов	2		2	Консультации
12	Пространства именования. Параметрический полиморфизм. Шаблоны. Шаблонные методы и классы. Основные типы библиотек	2		2	Консультации
13	Стандартные библиотеки и критерии их проектирования. Принципы дизайна и реализации стандартной библиотеки шаблонов языка Си++. Основные виды компонентов библиотеки шаблонов STL.	2		4	Консультации
14	Новые стандарты языка Си++	1		0	Консультации
Раздел 3. Основные компоненты системы программирования					
15	Основные компоненты систем программирования. Трансляторы, компиляторы, интерпретаторы. Общая схема работы компилятора	3		2	Консультации
Раздел 4. Элементы теории трансляции					
16	Основные понятия теории формальных языков: определение грамматик и языков, классификация по Хомскому. Задача разбора. Проблемы однозначности и эквивалентности грамматик. Алгоритм приведения грамматик	3		2	Консультации
1	**2**	**3**	**4**	**5**	**6**

17	Лексический анализ: регулярные грамматики и конечные автоматы. Алгоритм построения детерминированного автомата. Диаграммы состояний. Леволинейные и праволинейные грамматики	2		6	Консультации
18	Задачи лексического анализатора, схема его работы. Построение лексического анализатора по диаграмме состояний	2		4	Консультации
19	Синтаксический анализ по контекстно-свободным грамматикам. Метод рекурсивного спуска, его применимость	2		2	Консультации
20	Контроль контекстных условий. Семантический анализ описаний, выражений, операторов	2		2	Консультации
21	Внутреннее представление программ: примеры языков внутреннего представления. Польская инверсная запись (ПОЛИЗ). Синтаксически управляемый перевод	4		2	Консультации
22	Проблемы синтаксического и семантического анализа на примере модельного языка. Интерпретация внутреннего представления. Генерация кода	2		2	Консультации
23	Оптимизация в компиляторах: машинно-независимая оптимизация, машинно-зависимая оптимизация. Основные методы динамического распределения памяти	2		2	Консультации
24	Автоматизация построения лексических и синтаксических анализаторов	1		2	Консультации
25	Итоговая контрольная работа	1		4	Проверка письменной работы. Консультация
Всего, часов		**54**		**54**	
Итоговый контроль					**Экзамен - 36 часов**

где: Л – лекции, С – семинарские занятия, П – практические занятия, Лб – лабораторные занятия, СРС – самостоятельная работа студентов.

в)**. Содержание разделов дисциплины**

Планы лекций

ЛЕКЦИЯ № 1

1. Введение
2. Этапы жизненного цикла программного продукта
3. Технологии разработки программного обеспечения
4. Компоненты СП, поддерживающие разработчика программного обеспечения

Этапы жизненного цикла программного продукта. Иерархия программно-аппаратного обеспечения. Определение понятия "Система программирования". Программа, программный продукт, системный программный продукт. Жизненный (технологический) цикл программного продукта. Фаза разработки, фаза использования и фаза сопровождения.

Технологии разработки программного обеспечения (история и современность). Общая характеристика этапов разработки и сопровождения программных продуктов. Последовательность этапов разработки во времени. Нисходящая, каскадная, итеративные схемы. Каскадно-возвратный метод разработки. Спиральная модель жизненного цикла.

Компоненты СП, поддерживающие разработчика. Основные требования к системам программирования. Требования к составу систем программирования. Требования к инструментарию отдельных этапов разработки программных продуктов. Пример системы программирования операционной системы UNIX. Пример комплексной системы программирования Rational Software Corporation.

ЛЕКЦИЯ № 2

5. ООП – новая технология (парадигма) программирования
6. Основные принципы ООП

ООП – новая технология (парадигма) программирования. Понятие объекта в программировании. Процессно-ориентированный и объектно-ориентированный подходы к программированию. Абстракция, инкапсуляция, наследование, полиморфизм.

ЛЕКЦИЯ № 3

7. Обзор средств языка Си++, поддерживающих основные механизмы

Обзор средств языка Си++, поддерживающих эти механизмы. Язык Си++ в сравнении с языком Си. Понятие класса в языке Си++. Отображение основных принципов объектно-ориентированного программирования в языке Си++ на простейших примерах.

ЛЕКЦИЯ № 4

8. Виды отношений между классами
   - ✓ ER-диаграммы
   - ✓ Взаимодействие и иерархия классов
   - ✓ Примеры типизации, наследования и полиморфизма в Си++

Основные понятия модели Сущность-Связи (Entity-Relationship, ER). Ассоциация классов, агрегация классов, использование одним классом другого класса, инстанцирование класса, наследование одним классом свойств другого класса.

ЛЕКЦИЯ № 5

9. Приёмы декомпозиции
   - ✓ Объектно-ориентированный анализ предметной области

- ✓ Отображение сущностей предметной области в классы
- ✓ Преимущества объектной модели

10. Концепция абстрактных типов данных и её реализация в Си++

Выделение используемых объектов, фиксация связей между объектами, фиксация методов обмена сообщениями между объектами. Пример построения класса объектов. Интерфейс класса.

Концепция абстрактных типов данных и её реализация в Си++. Классы как средство создания новых типов.

ЛЕКЦИЯ № 6

11. Классы и объекты
12. Работа с состоянием объекта: селекторы и модификаторы (*getters/setters*)

*Синтаксис объявления класса. Ограничения доступа к элементам класса. Ключевое слово **inline**. Работа с состоянием объекта: селекторы и модификаторы. Процедуры и функции для установки или выдачи значения полей данных класса.*

ЛЕКЦИЯ № 7

13. Указатели и ссылки на объекты, передача параметров по ссылке
14. Инициализация, конструкторы, копирование, деструкторы

Указатели и ссылки на объекты. Указатель как обобщённый адрес объектов программы. Указатели на константы, константные указатели и константные указатели на константы. Ссылка на объект и её отличие от указателя. Передача параметров по ссылке.

*Определения и примеры использования конструкторов, операций копирования и деструкторов. Конструкторы умолчания. Автоматическая генерация конструкторов и деструкторов. Указатель **this**. Последовательность выполнения конструкторов и деструкторов для сложных объектов.*

ЛЕКЦИЯ № 8

15. Пример конструирования класса комплексных чисел
16. Работа с динамической памятью
17. Статические и константные члены классов

*Операции **new** и **delete**. Создание и уничтожение массивов объектов. Плоские и неплоские классы.*

*Квалификатор **const**. Статические методы. Инициализация статических членов класса.*

ЛЕКЦИИ № 9-10

18. Статический полиморфизм в Си++
    - ✓ Перекрытие и перегрузка имён
    - ✓ Перегрузка функций
    - ✓ Друзья классов
    - ✓ Перегрузка бинарных операций
    - ✓ Перегрузка унарных операций

Статический полиморфизм в Си++. Перекрытие имён. Перегрузка функций. Правила работы алгоритма выбора перегруженной функции. Перегрузка бинарных и унарных операций. Пример перегрузки операции индексирования. Особенности перегрузки префиксных и постфиксных операций.

ЛЕКЦИИ № 11-12

19. Одиночное наследование
20. Виды полиморфизма в Си++. Динамический полиморфизм
21. Виртуальные функции

Одиночное (единичное) наследование. Различия между наследованием и агрегацией. Правила наследования. Иерархия классов при наследовании. Решётки смежности. Повторное использование кода. Видимость при наследовании. Правила видимости. Перекрытие имён. Инициализация объектов внутри объектов. Список инициализации.

Генеалогические преобразования указателей.

Динамический полиморфизм в Си++. Виртуальные функции. Правила создания виртуальных функций. Виртуальные деструкторы.

ЛЕКЦИЯ № 13

22. Множественное наследование в Си++
23. Абстрактные классы. Чистые виртуальные функции
24. Фабрики объектов
25. Реализация виртуальных функций

Множественное наследование в Си++. Доступ к членам производного класса. Преобразование указателей. Виртуальные базовые классы. Неоднозначность из-за совпадающих имён в различных базовых классах.

Абстрактные классы. Чистые виртуальные функции. Позднее (динамическое) связывание.

Фабрики объектов. "Виртуальные" конструкторы.

Реализация виртуальных функций с помощью статических таблиц классов.

ЛЕКЦИЯ № 14

26. Средства обработки ошибок. Исключения в Си++
27. Динамическая идентификация типа, средства преобразования типов

Средства обработки ошибок. Исключения в Си++. Фундаментальная идея обработки ошибок времени выполнения в программах на Си++. Виды реакции на возникновение ошибки в программе. Управление исключительными ситуациями. Действия при возбуждении исключительной ситуации. Вложенные блоки обработчиков исключений. Стандартные исключительные ситуации.

Динамическая идентификация типа, средства преобразования типов. Динамическое и статическое приведение типа. Особенности приведения указателей и ссылок. Другие виды преобразования типов.

ЛЕКЦИЯ № 15

28. Пространства именования
29. Параметрический полиморфизм в Си++. Программирование с помощью шаблонов
30. Основные типы библиотек
31. Критерии проектирования стандартных библиотек

*Оператор разрешения области видимости. Видимость пространств именования. Директива **using**. Неименованные пространства именования. Отличия классов и пространств именования. Глобальные объекты.*

Шаблоны функций. Перегрузка шаблонных функций. Шаблоны классов. Паттерны метапрограммирования.

Основные типы библиотек. Библиотеки функций, процедур и макроопределений. Библиотеки классов и шаблонов. Библиотеки компонентов. Исполняющая среда. Критерии проектирования стандартных библиотек по составу и функциональности.

ЛЕКЦИЯ № 16

32. Стандартная библиотека языка Си++
33. Принципы дизайна и реализация стандартной библиотеки шаблонов Си++
34. Основные виды компонентов библиотеки шаблонов STL

Состав и функциональность стандартной библиотеки Си++.

Принципы дизайна и реализация стандартной библиотеки шаблонов Си++.

Основные виды компонентов STL: контейнеры, итераторы, алгоритмы, аллокаторы.

Векторы, строящиеся на основе контейнеров класса vector. Списки, строящиеся на основе стандартного контейнера list.

Достоинства и недостатки STL-подхода.

Перспективы новых стандартов языка Си++

ЛЕКЦИЯ № 17

35. Достоинства и недостатки STL-подхода
36. Новые стандарты языка Си++
37. Классическая система программирования
38. Трансляторы, компиляторы, интерпретаторы
39.Общая схема работы компилятора

Состав и схема функционирования классической системы программирования. Краткая характеристика отдельных компонентов СП, их роль и положение в общей схеме СП.

Типы трансляторов, особенности интерпретаторов и компиляторов. Схемы работы трансляторов. Интерпретаторы и компиляторы. Смешанная стратегия трансляции. Общая схема работы компилятора. Основные этапы компиляции. Однопроходный компилятор.

ЛЕКЦИЯ № 18

40.Основные понятия теории формальных языков
41.Классификация грамматик и языков по Хомскому
42.Задача разбора, дерево вывода
43. Проблемы однозначности и эквивалентности грамматик

Основные понятия теории формальных языков. Определения алфавита, цепочки, языка. Определение порождающей грамматики. Выводимость цепочек. Определение языка, порождаемого грамматикой. Эквивалентность грамматик.

Определение типов грамматик и языков по Хомскому. Соотношения между типами грамматик. Соотношения между типами языков. Примеры грамматик и языков.

Вывод. Цепочки вывода. Сентенциальные формы. Дерево вывода.

Неоднозначность вывода и грамматики. Неоднозначность языка. Проблемы эквивалентности грамматик.

ЛЕКЦИЯ № 19

44.Преобразования грамматик, группы и алгоритмы преобразования
45.Регулярные грамматики: разбор цепочек символов языка, построение конечного автомата по регулярной грамматике, построение грамматики по конечному автомату

Преобразования грамматик. Определение недостижимых и бесполезных символов, определение приведённой грамматики.

Разбор цепочек символов языков на основе регулярных грамматик. Определение детерминированного конечного автомата. Определение языка, заданного конечным автоматом. Алгоритм построения конечного автомата на основе леволинейной грамматики. Алгоритм построения леволинейной грамматики на основе описания конечного автомата. Диаграммы состояний. Построение диаграммы состояний. Анализаторы.

ЛЕКЦИЯ № 20

46.Лексический анализ на основе регулярных грамматик

- ✓ Недетерминированный конечный автомат, недетерминированный разбор
- ✓ Алгоритм преобразования недетерминированного конечного автомата в детерминированный
- ✓ Праволинейные грамматики
- ✓ Примеры преобразований конечных автоматов

Определение недетерминированного конечного автомата. Построение детерминированного конечного автомата по недетерминированному конечному автомату. Праволинейные грамматики.

ЛЕКЦИЯ № 21

47. Лексический анализ на основе регулярных грамматик (продолжение)

- ✓ Задачи лексического анализа на основе регулярных грамматик
- ✓ Иерархия классов лексического анализатора модельного языка
- ✓ Схема работы лексического анализатора модельного языка

Элементы теории трансляции. Лексический анализ на основе регулярных грамматик. Определение границ лексем. Обнаружение и распознавание лексем. Описание модельного языка. Лексический анализ на примере модельного языка. Объектно-ориентированный анализ и проектирование иерархии классов лексического анализатора. Разработка диаграммы состояний. Программирование по диаграмме состояний.

ЛЕКЦИЯ № 22

48. Задачи и проблемы синтаксического анализа
49. Метод рекурсивного спуска
50. Альтернативные методы распознавания контекстно-свободных языков

Задачи и проблемы синтаксического анализа.

Метод рекурсивного спуска: назначение, семантика процедур метода рекурсивного спуска. Достаточные условия применимости метода рекурсивного спуска. Подкласс грамматик, к которому применим метод. Расширение класса грамматик, к которым применим метод рекурсивного спуска (проблемы, преобразования, эквивалентность предлагаемых преобразований). О применимости метода рекурсивного спуска к модельному языку. Обработка ошибок при рекурсивном спуске.

Альтернативные методы распознавания контекстно-свободных языков. Алгоритмы с возвратами, табличные методы анализа. Алгоритмы анализа с линейным временем работы.

ЛЕКЦИЯ № 23

51. Задачи семантического анализа
52. Семантический анализ описаний, выражений и операторов

Задачи семантического анализа. Проверка семантических соглашений и контекстных условий. Дополнение внутреннего представления. Проверка правил хорошего программирования на языке. Распределение используемых в программе идентификаторов по пространствам именования.

Семантический анализ описаний. Грамматики с действиями. Контроль контекстных условий в выражениях. Контроль типов в операторах.

ЛЕКЦИЯ № 24

53. Внутреннее представление программ, свойства языков внутреннего представления
54. ПОЛИЗ выражений, алгоритм интерпретации ПОЛИЗ
55. ПОЛИЗ операторов языков программирования

Свойства языка внутреннего представления программы, примеры таких языков. Связные списочные структуры, тетрады, триады. Инфиксная, префиксная и постфиксная виды записи. Язык ассемблера и машинные команды.

Формальное определение ПОЛИЗ выражений. Достоинства инверсной записи выражений. Алгоритм интерпретации ПОЛИЗ.

ПОЛИЗ операторов языков программирования. Оператор присваивания. Оператор перехода. Условные операторы и операторы цикла. Операторы ввода и вывода модельного языка.

Синтаксически управляемый перевод: идея, принципы организации, примеры.

ЛЕКЦИЯ № 25

56.Синтаксически управляемый перевод
57.Синтаксический и семантический анализатор модельного языка
58. Генерация ПОЛИЗ программы на модельном языке

Синтаксический и семантический анализ модельного языка. Проектирование объектной модели анализатора. Семантический анализ описаний и выражений. Контроль типов в операторах.

Генерация ПОЛИЗ программы на модельном языке. Расширение объектной модели анализатора средствами генерации. Использование исключений C++ при обработке ошибок периода выполнения

Интерпретация ПОЛИЗ программы на модельном языке. Программа интерпретации для модельного языка. Генерация кода как альтернатива непосредственной интерпретации.

ЛЕКЦИЯ № 26

59. Интерпретация ПОЛИЗ программы на модельном языке
60. Генерация кода
61. Оптимизация в компиляторах

Основные оптимизирующие преобразования машинно-независимой оптимизации. Линейные участки. Оптимизация передачи параметров и вызовов функций. Оптимизация циклов. Пример оптимизирующих преобразований. Основные оптимизирующие преобразования машинно-зависимой оптимизации.

ЛЕКЦИЯ № 27

62. Основные методы динамического распределения памяти
63. Компоненты классической системы программирования
64. Автоматизация построения лексических анализаторов (LEX)
65. Автоматизация построения синтаксических анализаторов (YACC)

Основные методы явного и неявного динамического распределения памяти. Технологии динамического распределения памяти. Определение занятости блока памяти. "Сборка мусора".

Компоненты классической СП и их роль в общей системе. Интегрированная среда разработки. Причины возникновения, состав, основные возможности. Редакторы текстов.

Основные функции редактора текстов в рамках ИСР. Лексический анализ “на лету” – одна из функций текстового редактора. Библиотеки. Статические и динамически подключаемые библиотеки. Редакторы связей. Загрузчики. Средства конфигурирования. Системы управления версиями программных комплексов. Средства отладки и тестирования программ. Профилировщики. Справочные системы.

Автоматизация построения лексических анализаторов (Lex). Регулярные множества. Свойства регулярных выражений. Построение лексических анализаторов по регулярным выражениям. Автоматизация построения синтаксических анализаторов (Yacc).

5. Рекомендуемые образовательные технологии

Работа в аудитории: лекции; консультации перед экзаменом.

Процесс изложения учебного материала сопровождается презентациями и демонстрацией решения задач в интерактивном режиме.

Параллельно с чтением лекций организована учебная практика (Практикум на ЭВМ) студентов в объеме 108 часов. На учебной практике студенты выполняют индивидуальные задания, которые предназначены для закрепления теоретической части курса и получения практических навыков их применения.

Внеаудиторная работа: изучение пройденных на лекциях тем, самостоятельное изучение литературы по системам программирования

6. Учебно-методическое обеспечение самостоятельной работы студентов

Самостоятельная работа по изучению данной дисциплины включает:

- проработку теоретических основ лекционного материала;
- систематизацию изученного материала по курсу;
- научно-исследовательскую работу учащегося в библиотеках;
- подготовку к зачёту.

7. Оценочные средства для текущего контроля успеваемости

По итогам освоения дисциплины может проводиться письменная контрольная работа, которая предшествует зачёту, по результатам проверки контрольной работы производится разбор ошибок на консультации.

а) Образец письменной контрольной работы

«Системы программирования»
Вариант 1 ФИО_____ № группы_____

1	2	3	4	5	6	7	8	9	10

1. Что будет выдано в стандартный канал вывода при работе следующей программы?

```
class X;
void  F (X& x, int n);
class X
{ public: X ()    { try { F (* this, -2);           cout <<  1 << endl; }
                    catch (X&)    {                 cout <<  2 << endl; }
                    catch (int n) { F (* this, -n); cout <<  3 << endl; }
                  }
          X (X&)  {                                 cout <<  4 << endl; }
         ~X ()    {                                 cout <<  5 << endl; }
};
void F (X& x, int n) { try { if (n < 0) throw x;
                             if (n > 1) throw n;    cout <<  6 << endl;
                           }
                     catch (int) {                   cout <<  7 << endl;   throw;
}
                     catch (X&)  {                   cout <<  8 << endl;   throw;
}
}
void main()
{ try { X a; }
  catch (...)  {                                    cout <<  9 << endl;  }
                                                    cout << 10 << endl;
}
```

Ответ: _____

2. Дать определение класса, являющегося другом некоторого другого класса. Привести пример класса, имеющего дружественный класс, описание этого друга, и пример его использования.

3. STL: Написать функцию, которая считает количество положительных элементов заданного контейнера-списка `list<int>`, а затем распечатывает это значение (выдает в стандартный поток `cout`).

4. Дать определение абстрактного класса. Привести небольшой пример, поясняющий использование абстрактных классов в программе на C++.

5. Есть ли ошибки в тексте приведенной программы? Можно ли исправить определение класса и функции, чтобы программа стала верной? Если да, то как?

```
#include <iostream.h>
class A { public: int y; void f (int t) { cout << "f\n"; } };
g ()
{ const A a;
  a.f(A::y);
  return 0;
}
```

6. Дана грамматика G:

$S \to Bc$
$S \to dSc$
$dBc \to dAc$
$dA \to dAd$
$A \to d$

(а) Классификация: найти такое целое k, что G является грамматикой типа k и не является грамматикой типа k+1.
Ответ:

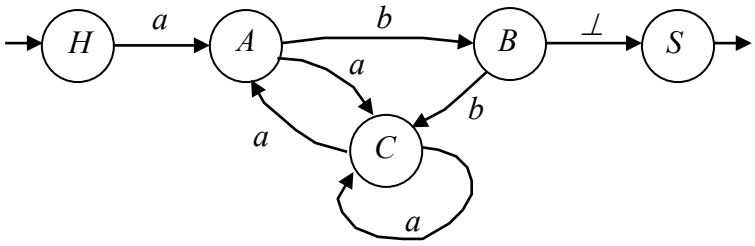
(б) Каким из перечисленных классов принадлежит язык $L(G)$?

Ответ:

Класс $\mathfrak{I}$	$L(G) \in \mathfrak{I}$? (да/нет)
контекстно-свободные языки	
контекстно-зависимые языки	
регулярные языки	
языки типа 0	

(в) Определение неукорачивающей грамматики. **Ответ**:

7. Конечный автомат $\mathcal{A}$ задан в виде диаграммы состояний:



(а) Показать, что автомат $\mathcal{A}$ не является детерминированным.

(б) По автомату $\mathcal{A}$ с помощью соответствующего алгоритма построить эквивалентный детерминированный автомат $\mathcal{A}_D$.

(в) По автомату $\mathcal{A}_D$ построить эквивалентную леволинейную грамматику G.

8. Дана грамматика:

G_1 : $S \to cAb$
$A \to dS \mid cd$

Вставить в G_1 действия вида $\langle cout << \ldots \rangle$ по переводу цепочек языка $L_1 = L(G_1)$ в цепочки языка $L_2 = \{ b(a)^{2k} \mid k \geq 1 \}$ так, чтобы символов a в цепочке из L_2 было в два раза больше, чем символов d
в соответствующей цепочке из L_1.

Ответ:

9. Записать в постфиксной записи оператор программы на Си++:

```
if (i > 0) x = j; else if (j > 0) x = i; else if (i < j) x = j;
```

Ответ:

ПОЛИЗ	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	

10. Какие виды отношений между классами поддерживают большинство объектно-ориентированных языков программирования? Коротко охарактеризуйте их.

б) Система итогового контроля знаний

По итогам освоения дисциплины проводится **устный зачёт**, по итогам которого выставляется оценка по пятибалльной системе.

Список вопросов к зачёту

1. Абстрактные типы данных, инкапсуляция, наследование, полиморфизм.
2. Класс, объект, состояние объекта, поведение объекта.
3. Си++: Пространства имен. Пространство имен *std*.
4. Си++: Конструкторы и деструкторы.
5. Си++: Присваивание и инициализация.
6. Си++: Ссылки в Си++. Передача параметров по ссылке.
7. Си++: Манипуляции с состоянием объекта.
8. Си++: Работа с динамической памятью.
9. Си++: Друзья класса.
10. Си++: Статические члены класса.
11. Виды полиморфизма в Си++ (статический, динамический, параметрический).
12. Си++: Статический полиморфизм. Перегрузка бинарных операций:
13. с помощью функции-члена класса
14. с помощью функции-друга класса
15. Си++: Статический полиморфизм. Перегрузка унарных операций:
16. с помощью функции-члена класса
17. с помощью функции-друга класса
18. Си++: Специфика перегрузки операций инкремента и декремента, операции индексации.
19. Си++: Статический полиморфизм. Перегрузка функций.
20. Си++: Алгоритм поиска оптимально отождествляемой функции.
21. Си++: Средства обработки ошибок. Исключения и обработка исключений.
22. Виды отношений между классами (ассоциация, наследование, агрегация, использование).
23. Си++: Одиночное наследование. Правила наследования. Видимость при наследовании.
24. Си++: Динамический полиморфизм. Виртуальные функции.
25. Принципы реализации виртуальных функций
26. Си++: Абстрактные классы.
27. Си++: Множественное наследование. Видимость при множественном наследовании. Виртуальные базовые классы.
28. Си++: Динамическая информация о типе (RTTI).
29. С++: Параметрический полиморфизм. Шаблонные функции.
30. С++: Шаблонные классы.
31. Стандартная библиотека Си++.
32. Стандартная библиотека шаблонов STL.
33. STL: контейнеры, итераторы, алгоритмы, аллокаторы.
34. Этапы жизненного цикла программного продукта.
35. Состав и схема функционирования классической системы программирования.
36. Типы трансляторов, особенности интерпретаторов и компиляторов. Смешанная стратегия трансляции.
37. Общая схема работы компилятора.
38. Основные понятия теории формальных грамматик и языков.
39. Эквивалентные грамматики.

40. Классификация формальных грамматик и языков по Хомскому.
41. Соотношения между типами грамматик.
42. Соотношения между типами языков.
43. Разбор цепочек по КС-грамматикам. Задача разбора. Дерево вывода.
44. Неоднозначность грамматик и языков.
45. Недостижимые и бесплодные символы грамматики. Алгоритмы удаления недостижимых и бесплодных символов. Приведенная грамматика. Алгоритм приведения грамматики.
46. Алгоритм удаления правил с пустой правой частью.
47. Определение недетерминированного конечного автомата (НКА).
48. Диаграмма состояний (ДС) конечного автомата.
49. Леволинейные регулярные грамматики и конечные автоматы.
50. Определение детерминированного конечного автомата (ДКА).
51. Алгоритм построения детерминированного конечного автомата по НКА.
52. Задачи лексического анализа.
53. Лексический анализ на основе регулярных грамматик.
54. Задачи синтаксического анализа.
55. Метод рекурсивного спуска (РС-метод): назначение, семантика процедур метода рекурсивного спуска.
56. Достаточные условия применимости метода рекурсивного спуска для грамматик без ε-альтернатив и для грамматик с ε-альтернативами.
57. Критерий применимости РС-метода.
58. Задачи семантического анализа. Грамматики с действиями.
59. Свойства языка внутреннего представления программы, примеры таких языков.
60. Синтаксически управляемый перевод: идея, принципы организации, примеры. Определение формального перевода.
61. ПОЛИЗ выражений.
62. ПОЛИЗ операторов языков программирования.
63. Генерация ПОЛИЗа выражений и операторов.
64. Интерпретация ПОЛИЗа.
65. Основные стратегии распределения памяти.
66. Машинно-независимая и машинно-зависимая оптимизация. Примеры оптимизирующих преобразований.
67. Интегрированная среда разработки программного обеспечения (ИСР).
68. Основные функции редактора текстов в рамках ИСР. Примеры его интегрированности с другими компонентами ИСР.
69. Отладчики, их возможности. Примеры интегрированности отладчика с другими компонентами ИСР.
70. Редактор внешних связей, его назначение и принципы работы. Загрузчик.
71. Профилировщики. Назначение. Принцип работы.
72. Библиотеки. Основные типы библиотек.
73. Критерии проектирования стандартных библиотек.
74. Способы подключения библиотек функций.

8. Учебно-методическое и информационное обеспечение дисциплины

а) основная литература

1. И. А. Волкова, А. В. Иванов, Л. Е. Карпов. Основы объектно-ориентированного программирования. Язык программирования C++. Учебное пособие для студентов 2 курса. — М.: Издательский отдел факультета ВМК МГУ, 2011.
   Электронная версия: http://cmcmsu.info/download/cpp.base.oop.pdf
2. И. А. Волкова, А. А. Вылиток, Л. Е. Карпов. “Задачи и упражнения по языку Си++”, МГУ, МАКС Пресс, 2013
   Электронная версия: http://cmcmsu.info/download/cpp.tasks.2013.pdf
3. И. А. Волкова, А. А. Вылиток, Т. В. Руденко. Формальные грамматики и языки. Элементы теории трансляции (3-е издание). — М.: Изд-во МГУ, 2009.
   Электронная версия:
   http://cmcmsu.info/download/formal.grammars.and.languages.2009.pdf
4. И. А. Волкова, И. Г. Головин, Л. Е. Карпов. Системы программирования (Учебное пособие) . — М.: Издательский отдел факультета ВМиК МГУ, 2009.
   Электронная версия: http://cmcmsu.info/download/programming.systems.course.pdf
5. И. Г. Головин. Практикум на ЭВМ. Модельный веб-сервер: Методическое пособие для студентов II курса. — М.: Издательский отдел факультета ВМиК МГУ, 2009
   Электронная версия: http://cmcmsu.info/download/cpp.http.server.pdf
6. И. А. Волкова, И. Г. Головин, Л. Н. Кузина, М. Г. Мальковский. Модельный SQL-интерпретатор. — М.: Изд-во МГУ, 2005.
   Электронная версия: http://cmcmsu.info/download/model.sql.interpreter.2005.pdf
7. Ю.С. Корухова. Сборник задач и упражнений по языку C++. Учебное пособие для студентов-бакалавров II курса, обучающихся по направлению «Информационные технологии». — М.: Издательский отдел факультета ВМК МГУ, 2009
   Электронная версия: http://cmcmsu.info/download/korukhova.cpp.tasks.pdf
8. Л. Е. Карпов "Архитектура распределённых систем программного обеспечения", М.: МАКС Пресс, МГУ, ВМ и К, 132 стр., 2007 г. (шифр в библиотеке МГУ: 5ВГ66, К-265). *Электронная версия*: http://sp.cmc.msu.ru/courses/sdpi/mdwrbook.pdf

б) дополнительная литература:

1. Д. Грис. Конструирование компиляторов для цифровых вычислительных машин. — М.: Мир, 1975.
2. Ф. Льюис, Д. Розенкранц, Р. Стирнз. Теоретические основы проектирования компиляторов. — М.: Мир, 1979.
3. А. Ахо, Дж. Ульман. Теория синтаксического анализа, перевода и компиляции, тт.1, 2 — М.: Мир, 1979.
4. Л. Бек. Введение в системное программирование. — М.: Мир, 1988.
5. А. Ахо, Р. Сети, Дж. Ульман. Компиляторы: принципы, технологии, инструменты. — М.: Изд. дом «Вильямс», 2001. (Шифр в библиотеке МГУ: 5ВГ66 А-955)
6. А. В. Гордеев, А. Ю. Молчанов. Системное программное обеспечение. — СПб.: Питер, 2001
7. Молчанов А.Ю. Системное программное обеспечение. – СПб, Питер, 2003.

8. Г. Буч. Объектно-ориентированный анализ и проектирование с примерами приложений на C++, 2-е издание. — М. СПб.: «Издательство Бином» — «Невский диалект»,1998.
9. А. Элиенс. Принципы объектно-ориентированной разработки программ, 2-е издание. — М.: Издательский дом «Вильямс», 2002. (Шифр в библиотеке МГУ: 5ВГ66 Э-460)
10. И. О. Одинцов. Профессиональное программирование. Системный подход. — СПб.: БХВ-Петербург, 2002. (Шифр в библиотеке МГУ: 5ВГ66 О-425)
11. Н. Н. Мансуров, О. Л. Майлингова. Методы формальной спецификации программ: языки MSC и SDL. — М.: Изд-во «Диалог-МГУ», 1998. (Шифр в библиотеке МГУ: 5ВГ66 М-238)
12. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. – М. Финансы и статистика, 2000.
13. А. М. Вендров. CASE-технологии. Современные методы и средства проектирования информационных систем. — Электронная публикация на CITFORUM.RU
14. М. Фаулер, К. Скотт. UML в кратком изложении. Применение стандартного языка объектного моделирования. — М.: Мир, 1999. (Шифр в библиотеке МГУ: 5ВГ66 Ф-282)
15. Г. Майерс. Искусство тестирования программ. — М.: «Финансы и статистика», 1982
16. С. Канер, Дж. Фолк, Е. К. Нгуен. Тестирование программного обеспечения. — М.: «DiaSoft», 2001
17. Дж. Макгрегор, Д. Сайкс. Тестирование объектно-ориентированного программного обеспечения. Практическое пособие. — М.: «DiaSoft», 2002
18. Б. Страуструп. Язык программирования C++. Специальное издание. — М.: Издательство «БИНОМ», 2001. (Шифр в библиотеке МГУ: 5ВГ66 С-835)
19. Г. Шилдт. Самоучитель C++. 3-е изд. — СПб: БХВ-Петербург, 2002. (Шифр в библиотеке МГУ: 5ВГ66 Ш-576).
20. Шилдт Г. MFC. Основы программирования. Дюссельдорф-Киев-Москва-Санкт-Петербург, 1997.
21. Паппас К., Мюррэй У. Visual C++. Руководство для профессионалов. – СПб, BHV, 1996.
22. Александреску А. Современное проектирование на C++. Обобщенное программирование и прикладные шаблоны проектирования. - М., Вильямс, 2002.
23. Брауде Э.-Дж. Технология разработки программного обеспечения. – СПб, Питер, 2004
24. Фридман А.Л. Основы объектно-ориентированной разработки программных систем. – М. Финансы и статистика, 2000.
25. Прайс Дж., Гандерлой М. Visual C# .NET. Полное руководство. – Киев, Век+, 2004.
26. Сван Т. Основы программирования в Delphi для Windows95 - Киев: Диалектика, 1996.

в) Интернет-ресурсы

- электронная учебно-методическая система «Ownlibrari» кафедры программирования Филиала МГУ в г. Севастополе
- сайт ВМК МГУ - http://cmcmsu.info/2course/

9. Материально-техническое обеспечение дисциплины

- библиотека Филиала МГУ в г. Севастополе;
- библиотека кафедры программирования Филиала МГУ в г. Севастополе;
- лекционная аудитория, оборудованная средствами подключения к сети электропитания и локальной сети университета, а также средствами интерактивного отображения информации для показа презентаций лекций и демонстрации решения задач;
- для самостоятельной работы студентов – специализированные компьютерные классы с доступом к Интернет-ресурсам с любого компьютера, а также с ресурсами способными поддерживать системы программирования **Microsoft Visual Studio** версии не ниже 2008, **eclipse** версии не ниже **INDIGO**.

ПРИЛОЖЕНИЕ А

ПОЛОЖЕНИЕ
о проведении пересдач задолженностей студентов по итоговым аттестациям на кафедре прикладной математики Филиала МГУ в г. Севастополе прикомандированными преподавателями с факультета ВМК МГУ протокол №5 заседания кафедры от 10 апреля 2012 г.

Настоящее положение регулирует порядок проведения пересдач задолженностей студентами факультета Компьютерной математики на кафедре программирования Филиала МГУ в г. Севастополе прикомандированными преподавателями с факультета ВМК МГУ.

1. Пересдача задолженностей принимается по графику пересдач задолженностей, установленному кафедрой программирования в принятые ректоратом МГУ имени М.В.Ломоносова сроки, в отдельных случаях – в сроки, установленные комиссией по студенческим делам.
2. График пересдач задолженностей сообщается в учебный отдел в виде служебной записки и преподавателю факультета ВМК МГУ, по дисциплине которого проводится пересдача.
3. Пересдачи экзаменов (зачётов) в отсутствие преподавателей факультета ВМК в Филиале МГУ в г. Севастополе проводятся в письменной форме, независимо от того в какой форме проводился основной экзамен (зачёт).
4. Вариант письменной экзаменационной (зачётной) работы для пересдачи составляет преподаватель, проводивший основной экзамен. Также преподаватель указывает требования к проведению экзамена (сколько времени даётся на написание работы, какими материалами разрешается пользоваться студенту при написании работы и др.)
5. Кафедрой программирования назначаются местные преподаватели кафедры, которые проводят пересдачу в установленные даты и время в соответствии с графиком пересдач по присланному варианту с выполнением всех требований к проведению.
6. Написанные студентами при пересдаче экзаменационные (зачётные) работы сканируются по окончании пересдачи и пересылаются по электронной почте на проверку преподавателю факультета ВМК МГУ.
7. Преподавателю ВМК МГУ пересылается скан ведомости пересдачи для выставления отметок по результатам пересдачи.
8. Преподаватель ВМК присылает в Филиал скан заполненной им ведомости пересдачи и сканы проверенных им работ.
9. Полученный скан ведомости подписывается зам. зав. кафедрой.
10. Скан ведомости пересдачи сдаётся в учебный отдел Филиала.
11. Кафедра знакомит студентов, писавших работу на пересдаче, с результатами пересдачи.
12. При возникновении у студентов, писавших работу на пересдаче, вопросов по результатам проверки, студент может обратиться к преподавателю ВМК лично по электронной почте (скайпу). Свои координаты для консультаций с ним преподаватель сообщает студентам на первой лекции курса.

Приложение

Пример экзаменационного билета

Специальность (направление) **Прикладная математика и информатика**

Направление 01.03.02. Прикладная математика и информатика

Учебная дисциплина **Системы программирования**

Семестр 4

Экзаменационный билет № 1

1. Фазы жизненного цикла программного продукта. Этапы разработки, последовательность выполнения работ при разработке программных продуктов.
2. Определение абстрактного типа данных. Пример реализации концепции абстрактного типа данных на языке Си++.

Утверждено на заседании кафедры,

протокол № ____ от « ___ » _____ 2019 г.

Зав. кафедрой _____ (С. И. Гуров)

Преподаватель _____ (Л. Е. Карпов)