Федеральное государственное бюджетное образовательное учреждение высшего образования
Московский государственный университет имени М.В. Ломоносова
Факультет Вычислительной математики и кибернетики
Кафедра Алгоритмических языков

УТВЕРЖДАЮ
Декан факультета ВМК МГУ
_____ /И.А.Соколов/
«24» 03 _____ 2023 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Наименование дисциплины (модуля):

Системы программирования

Уровень высшего образования:

бакалавриат

Направление подготовки (специальность):

02.03.02 Фундаментальная информатика и информационные технологии

Направленность (профиль) ОПОП:

Фундаментальная информатика и информационные системы

Форма обучения:
очная

Москва 2023

Рабочая программа дисциплины (модуля) разработана в соответствии с самостоятельно установленным МГУ образовательным стандартом (ОС МГУ) для реализуемых основных профессиональных образовательных программ бакалавриата, магистратуры, реализуемых последовательно по схеме интегрированной подготовки по направлениям 02.03.02, 02.04.02 «Фундаментальная информатика и информационные технологии» в редакции приказа МГУ от 30 декабря 2016г.

1. Место дисциплины (модуля) в структуре ОПОП ВО

Дисциплина относится к дисциплинам базовой части ОПОП ВО и является обязательной для освоения в 4-ом семестре обучения.

2. Входные требования для освоения дисциплины (модуля), предварительные условия:

Учащиеся должны владеть знаниями по алгоритмам и алгоритмическим языкам, архитектуре ЭВМ и языку Ассемблера, программированию на языке Си и основам объектно-ориентированного программирования на языке C++ в объеме, соответствующем программе первых трех семестров обучения основных образовательных программам бакалавриата по укрупненным группам направлений и специальностей 01.00.00 «Математика и механика», 02.00.00 «Компьютерные и информационные науки».

3. Результаты обучения по дисциплине (модулю), соотнесенные с требуемыми компетенциями выпускников.

Компетенции выпускников, формируемые (полностью или частично) при реализации дисциплины (модуля):

ОПК-1 - способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с фундаментальной информатикой и информационными технологиями;

ОПК-2 - способность применять в профессиональной деятельности современные языки программирования и языки баз данных, методологии системной инженерии, системы автоматизации проектирования, электронные библиотеки и коллекции, сетевые технологии, библиотеки и пакеты программ, современные профессиональные стандарты информационных технологий;

ОПК-3 - способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям;

ПК-6 - способность эффективно применять базовые математические знания и информационные технологии при решении проектно-технических и прикладных задач, связанных с развитием и использованием информационных технологий;

ПК-7 - способность разрабатывать и реализовывать процессы жизненного цикла информационных систем, программного обеспечения, сервисов систем информационных технологий, а также методы и механизмы оценки и анализа функционирования средств и систем информационных технологий;

ПК-9 - способность разрабатывать, оценивать и реализовывать процессы жизненного цикла информационных систем, программного обеспечения, сервисов информационных технологий, а также реализовывать методы и механизмы оценки и анализа функционирования средств и информационных технологий; разрабатывать проектную и программную документацию, удовлетворяющую нормативным требованиям;

Планируемые результаты обучения по дисциплине (модулю):

Знать:

- основные понятия языка С++;
- понятие перегрузки функций и операций;
- основные понятия наследования, единичное и множественное наследование;
- понятие механизма виртуальных функций;
- основные виды отношений между классами, ER-диаграммы;
- понятие об обработке исключений в С++;
- понятие о динамической идентификации типов;
- основы параметрического полиморфизма, шаблоны функций и классов;
- определение понятия "Системы программирования";
- основные требования к системам программирования;
- состав и схему функционирования классической системы программирования;
- основы теории формальных языков и грамматик;
- классификацию формальных грамматик и языков по Хомскому;
- основы теории трансляции;
- типы трансляторов и схемы их работы;
- задачи и схемы работы лексического, синтаксического и семантического анализаторов.

Уметь:

- создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов;
- самостоятельно разрабатывать алгоритмы для решения задач системного и прикладного программирования.
- применять метод преобразования недетерминированного конечного автомата к детерминированному;
- применять метод рекурсивного спуска к КС- грамматикам;
- применять на практике основные методы теории трансляции.

Владеть:

- навыками решения практических задач на языке С++;
- навыками решение задач по теории трансляции;
- навыками решения практических задач по теории формальных грамматик.

Иметь опыт:

- разработки компонентов программного обеспечения на основе использования объектно-ориентированной парадигмы программирования;
- решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.

4. Формат обучения

Работа в аудитории:
Используются традиционные технологии проведения лекций в аудиториях с использованием меловой доски, а также чтение лекций с использованием слайдов. При чтении лекций наиболее

важные элементы лекций обсуждаются с аудиторией в режиме «вопрос-ответ».
Лектор курса периодически проводит консультации по дисциплине.

Внеаудиторная работа:
изучение пройденных на лекциях тем, самостоятельное изучение литературы по изучаемому материалу.

5. Объем дисциплины (модуля) составляет 3 зачетные единицы, всего 108 часов,
в том числе 48 академических часов, отведенных на контактную работу обучающихся с преподавателем (лекции – 48 часов), 60 академических часа на самостоятельную работу обучающихся.

6. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и виды учебных занятий

Целью освоения дисциплины является знакомство с современными системами программирования, значительная часть которых базируется на объектно-ориентированных языках программирования. В частности, ставятся следующие задачи:

- продолжить изучение возможностей объектно-ориентированного языка программирования C++, начатого в 3 семестре в курсе «Объектно-ориентированное программирование»;
- изучить основы теории формальных языков и грамматик, необходимой при построении трансляторов – основного компонента систем программирования;
- познакомиться с современными системами программирования (состав, назначение, схема работы);

Наименование и краткое содержание разделов и тем дисциплины (модуля), **Форма промежуточной аттестации по дисциплине (модулю)**	**Всего** (часы)	В том числе			
		Контактная работа (работа во взаимодействии с преподавателем) Виды контактной работы, часы			**Самостоятельная работа обучающегося, часы** *(виды самостоятельной работы – эссе, реферат, контрольная работа и пр.)*
		Занятия лекционного типа*	Занятия семинарского типа*	**Всего**	
1.Обзор основных свойств и возможностей объектно-ориентированного программиро-вания на языке Си++, рассмотрен-ных в курсе “ООП” в 3 семестре. Работа с динамической памятью. Создание и уничтожение массивов объектов. Плоские и неплоские классы.	3	1	0	1	2
2. Статический полиморфизм в C++. Перегрузка функций и операторов. Перегрузка унарных операций с помощью функции-члена класса и с помощью функции-друга класса.	9	5	0	5	4

Правила перегрузки бинарных и унарных операций в C++. Особенности перегрузки префиксных и пост-фиксных операций инкремента и декремента, операции ->, операции индексирования, опера-ции присваивания, операции приведения к типу, операции ввода – вывода << и >>.					
3. Понятие наследования. Единичное наследование. Закрытое и защищенное (private, protected) наследование. Спецификатор доступа к членам класса protected. Функции-члены производного класса - наследование и замещение, правила видимости. Повторное использование кода. Конструкторы и деструкторы производных классов: порядок вызова, передача параметров. Указатели на базовый и произ-водный классы, преобразование указателей. Виртуальные функции. Абстрактные классы. Использование виртуальных деструкторов.	8	4	0	4	4
4. Множественное наследование в C++. Основные проблемы и способы их решения. Доступ к членам производного класса. Преобразование указателей. Виртуальные базовые классы. Неоднозначность из-за совпадающих имён в различных базовых классах.	4	2	0	2	2
5. Исключения в C++. Общая схема обработки исключений: try - throw – catch. Правило выбора обработчика исключения. Передача исключения в объемлющий блок (throw).	4	2	0	2	2
6. Динамическая идентификация типа, средства преобразования типов.	3	1	0	1	2
7. Виды отношений между классами. ER-диаграммы. Язык спецификаций. Ассоциации классов. Взаимодействие и иерархия классов. (Основные понятия модели Entity-Relationship. Ассоциация классов, агрегация классов, использование одним классом другого класса, инстанцирование класса, наследование одним классом свойств другого класса.)	3	1	0	1	2
8. Параметрический полимор-физм. Программирование с помощью шаблонов. Шаблоны функций и классов	4	2	0	2	2

.					
9. Определение понятия "Система программирования". Иерархия программно-аппаратного обеспечения. Программа, программный продукт, системный программный продукт. Этапы жизненного цикла программного продукта(фаза разработки, фаза использования и фаза сопровождения). Технологии разработки программного обеспечения. Общая характеристика этапов разработки и сопровождения программных продуктов. Последовательность этапов разработки во времени. Компоненты СП, поддерживающие разработчика. Основные требования к системам программирования. Требования к составу систем программирования. Пример системы программирования операционной системы UNIX. Классическая система программирования. Состав и схема функционирования классической системы программирования. Краткая характеристика отдельных компонентов СП, их роль и положение в общей схеме СП.	8	6	0	6	2
10. Компоненты классической системы программирования. Интегрированная среда разработки. Редакторы текстов. Библиотеки. Статические и динамически подключаемые библиотеки. Редакторы связей. Загрузчики. Средства конфигурирования. Системы управления версиями программных комплексов. Средства отладки и тестирования программ. Профилировщики. Справочные системы. Краткий обзор современных систем программирования, поддерживающих ООП, и их возможностей.	4	2	0	2	2
11. Типы трансляторов, особенности интерпретаторов и компиляторов. Схемы работы трансляторов. Интерпретаторы и компиляторы. Смешанная стратегия трансляции. Общая схема работы компилятора. Основные этапы компиляции. Однопроходный компилятор.	4	2	0	2	2

12. Основные понятия теории формальных грамматик. Определения алфавита, цепочки, языка. Определение порождаю-щей 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 грамма-тик и языков. Вывод. Цепочки вывода. Сентенциальные формы. Дерево вывода. Неоднозначность вывода и грамматики. Неоднозначность языка. Преобразования грамматик. Определение недостижимых и бесполезных символов, определение приведённой грамматики.	8	4	0	4	4
13. Регулярные и автоматные грамматики. Леволинейные и праволинейные регулярные грамматики. Алгоритм получения леволинейной грамматики по праволинейной. Диаграммы состояний. Построение диаграммы состояний. Определение детерминированного и недетерминированного конечного автомата. Построение детерминированного конечного автомата по недетерминиро-ванному конечному автомату.	6	4	0	4	2
14. Лексический анализ на основе регулярных грамматик. Задачи лексического анализа. Иерархия классов лексического анализатора модельного языка. Схема работы лексического анализатора модельного языка.	4	2	0	2	2
15. Метод рекурсивного спуска: назначение, семантика процедур метода рекурсивного спуска. Достаточные условия применимости метода рекурсивного спуска. Подкласс грамматик, к которому применим метод. Критерий применимости метода рекурсивного спуска.	4	2	0	2	2
16. Задачи и проблемы синтаксического анализа. Расширение класса грамматик, к которым применим метод рекурсивного спуска. Преобразования грамматик, эквивалентность рассматривае-мых преобразований. О	4	2	0	2	2

примени-мости метода рекурсивного спуска к модельному языку.					
17. Задачи семантического анализа. Грамматики с действия-ми. Контроль контекстных условий в выражениях. представление программ, свойства языков внутреннего представления. ПОЛИЗ выражений, алгоритм интерпретации ПОЛИЗ. Алгоритм Дейкстры. ПОЛИЗ операторов языков программирования. Оператор присваивания. Оператор перехода. Условные операторы и операторы цикла. Операторы ввода и вывода модельного языка. Синтаксически управляемый перевод.	4	2	0	2	2
18. Синтаксический и семантический анализатор модельного языка. Генерация ПОЛИЗ программы на модельном языке. Интерпретация ПОЛИЗ программы на модельном языке. Генерация кода.	4	2	0	2	2
19. Решение типовых задач по курсу. Подготовка к Коллоквиуму.	4	2	0	2	2
20.Текущий контроль успеваемости: коллоквиум	6	0	0	0	6
21. Промежуточная аттестация: экзамен.	10	0	0	0	10
Итого	108	48	0	48	60

7. Фонд оценочных средств (ФОС) для оценивания результатов обучения по дисциплине (модулю)

7.1. Типовые контрольные задания или иные материалы для проведения текущего контроля успеваемости.

Вариант письменного коллоквиума (образец) :

NB! Во всех задачах, где это требуется, предполагается наличие необходимых включений и директивы using namespace std.

1.Исправьте только описание структуры C так, чтобы в приведенной ниже программе не было ошибок, а на экран напечаталось f (int, int).

```
struct C { int n;
           C (int  m) { n = m; }
           operator int () { return 1; }
};
void  f ( int i,  int  j )   { cout <<  "f(int, int)\n";}
void  f ( C b,  C a )   { cout <<  "f(A, A)\n";}

int main () {                C a (1);
```

```
        f (a, 1);
        return 0;   }
```

2.Что такое абстрактный класс в С++? Обязательно приведите пример описания и использования абстрактного класса.

3.Укажите все прототипы конструкторов и деструкторов в порядке их выполнения в следующей программе:

```
class A {};
class B: public A{};
struct  D { A a; };

int main () {
        D d;
        { d.a = B();
          B b; }
        A a1, a2 = a1;
        return 0;
}
```

4.Что напечатает программа?

```
struct B {
        virtual void f (int a = 0) { cout << a << "B::f (int) \n";}
        void g () { f (); cout <<"B::g () \n";}
};

class D: public B {
                virtual void f () { cout << "D::f()";}
                virtual void g () { f (); cout << "D::g()\n";}
};

int main () {
                D d;
                B b, & rb = d;
                 b. g ();
                rb. g ();
                return 0;
          }
```

5. Что такое функция-друг в языке Си++? Приведите пример описания и использования функции-друга.

6.Добавить в приведенную программу необходимые конструкции С++ (не заменяя и не удаляя имеющиеся) так, чтобы при ее компиляции не было ошибок, а на печать выдалось:

Constr_B
Constr_A

Destr_A
Destr_B

```
struct B {
                virtual void f() = 0;
                B(){ cout << "Constr_B\n";}
               ~B(){ cout << "Destr_B\n"; }
      };
 int main () {
                NN::A a;
                B *pb = &a;
                return 0;
  }
```

7.Добавьте в следующую программу необходимые описания так, чтобы в ней не было ошибок.

```
      int main () {
                const A a;
                a.x = 3;
                cout << a.y << a.x << a.f(1) << endl;
                return 0; }
```

8. Модифицируйте функцию *main()*, ничего в ней не удаляя, не используя комментарии, goto и прочие переходы так, чтобы программа завершалась нормально (не аварийно).

```
struct B { virtual void f () { cout << "B::f()\n"; } };
struct D : B { void f () { cout << "D::f()\n"; } };

int main () {
        D d, &rd  = d;
        B b, & rb = b, &rbd = d;
        rd = dynamic_cast <D&> (rbd);  rd.f();
        rd = dynamic_cast <D&> (rb);   rd.f();
        return 0;
}
```

7.2. Типовые контрольные задания или иные материалы для проведения промежуточной аттестации.

Вопросы к экзамену

1. Основные понятия и определения теории формальных языков. Примеры.
2. Классификация формальных грамматик и языков по Хомскому. Примеры.
3 .Соотношения между типами грамматик, соотношения между типами языков. Эквивалентные и почти эквивалентные грамматики.
4. Бесплодные и недостижимые символы. Приведенные грамматики. Алгоритм приведения грамматики.
5. Вывод. Дерево вывода. Примеры.
6. Понятие неоднозначности грамматики и языка. Примеры.
7. Деревья вывода при разборе по регулярным грамматикам.

8. Регулярные грамматики и конечные автоматы. Алгоритмы построения грамматики по автомату и автомата по грамматике.
9. Детерминированные и недетерминированные конечные автоматы. Алгоритм преобразования НКА в ДКА.
10. Синтаксический анализ. Метод рекурсивного спуска.
11. Модификация метода рекурсивного спуска для грамматик с итерациями и для грамматик, содержащих epsilon-правила. Применимость метода к таким грамматикам.
12. Синтаксически управляемый перевод.
13. Грамматики с действиями. Контроль контекстных условий в выражениях.
14. ПОЛИЗ выражений, алгоритм интерпретации ПОЛИЗ. Алгоритм Дейкстры.
15. ПОЛИЗ операторов языков программирования.
16. Понятие системы программирования, её роль и место в составе вычислительной системы.
17. Состав и схема функционирования классической системы программирования.
18. Жизненный цикл программного продукта.
19. Компоненты классической системы программирования - редакторы текстов, библиотеки, редакторы связей, загрузчики, системы управления версиями программных комплексов, средства отладки и тестирования программ.
20. Понятия интерпретатора и компилятора, схемы их работы. Смешанная стратегия трансляции.
21.Объектно-ориентированное программирование (ООП) - новая парадигма программирования. ООП-анализ.
22.Пространства имен в языке Си++.
23.Понятие класса и объекта. Описание класса.
24.Управление доступом к членам класса -- public, private, protected.
25.Объявления и описания функций-членов класса; эффект inline.
26.Специальные функции -- конструкторы и деструктор.
27. Конструктор копирования. Конструктор копирования и операция присваивания: содержательная связь и различие.
28. Ссылки и указатели в Си++.
29. Операторы new и delete.
30. Друзья класса, "законы" дружбы. Сравнение функции-члена и функции-друга: описание, вызов.
31. Статические члены класса.
32. Константные функции-члены.
33. Функции с параметрами по умолчанию.
34. Перегрузка функций. Перегрузка и неоднозначность.
35. Алгоритм поиска оптимально отождествляемой (best-matching) функции
36. Перегрузка операторов. Перегрузка с помощью функции-члена и функции-друга.
37. Перегрузка бинарных операций в С++.
38. Перегрузка унарных операций в С++.
39. Особенности перегрузки операций ++ и --, операции индексации в С++.
40. Особенности перегрузки операции присваивания.
41. Особенности перегрузки операции - > и приведения типов.
42. Особенности перегрузки операций ввода /вывода (>> и <<).
43. Обработка исключений в С++. Преобразование типов в обработчиках исключений.
44.Одиночное наследование. Правила наследования. Видимость при наследовании.
45. Конструкторы и деструкторы при наследовании.
46. Указатели на базовый и производный классы, преобразование указателей.

47. Динамический полиморфизм. Виртуальные функции.
48. Чисто виртуальные функции. Абстрактные классы.
49. Множественное наследование в C++. Основные проблемы и способы их решения.
50. Динамическая информация о типе (RTTI).
51. Виды отношений между классами. ER-диаграммы. Язык спецификаций.
52. Параметрический полиморфизм. Шаблонные функции
53. Параметрический полиморфизм. Шаблонные классы.

Типовые задачи для экзамена

Задача №1.
Вычеркните неразрешимые вызовы функции f (...). Что напечатает получившаяся программа?

```
struct A {
            operator int () { return 1; }
          };
void f (double d, char c) { cout << "f (double, char) \n";}
void f (double d, int j) { cout << "f (double, int) \n";}
void f (A a, const char * p) { cout << "f (A, const char *) \n";}
void f (int i, const char * p) { cout << "f (int, const char *) \n";}

int main () {
              A a ;
              f (a, 0);
              f (a, 'a');
              f ('a', 0);
              return 0;
       }
```

Задача №2.
Исправьте ошибки в нижеследующей программе, ничего в ней не удаляя. Объясните, в чем заключаются исправленные ошибки. Что напечатает программа после внесенных исправлений?

```
struct A {
            int y;
            int f () { return (y = 2); }
       };

struct B : A {
              static int g () { return 9 + f(); }
             };

 int A::y;
 int main () {
              B b, d;
              b.y = b.f ();
              cout << B::f() << ' ' << B::y << ' ' << d.y << B::g() << endl;
              return 0;
    }
```

Задача №3.
Опишите структуру с именем S, удовлетворяющую двум условиям:
-- можно создать объект типа S;
-- нельзя создать массив элементов типа S в динамической памяти.

Задача №4.
Даны грамматики G_1 и G_2.

G_1: $S \to ASB \mid \varepsilon$
$AB \to BA$
$A \to a$
$B \to b$

G_2: $S \to aSbS \mid bSaS \mid \varepsilon$

а) Эквивалентны ли G_1 и G_2? Ответ обосновать.
б) Определить тип грамматики G_1 :
в) Определить тип грамматики G_2 :
г) Определить тип языка $L(G_1)$:
д) Определить тип языка $L(G_2)$

Задача №5.
Дана леволинейная грамматика G_{left} :

$S \to Sa \mid Aa$
$A \to Aa \mid b \mid a$

а) Применим ли метод рекурсивного спуска к данной грамматике? Обоснуйте ответ.

б) Постройте эквивалентную детерминированную праволинейную грамматику G_{right}

Задача №6.
Что такое оптимизация программы? Ниже дан ПОЛИЗ фрагмента программы на языке Си. Оптимизируйте длину кода (постройте более короткий эквивалентный ПОЛИЗ), используя только те виды операций, которые есть в исходном ПОЛИЗе.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
ПОЛИЗ	*a*	*b*	>	12	!F	*&a*	c	=	;	16	!	&a	*d*	=	;		

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

Задача №7.
Модифицируйте программу, ничего в ней не удаляя, так, чтобы операция dynamic_cast работала безошибочно, а программа в целом завершилась нормально (не аварийно).

```
struct B {
        void g1 () { cout << "B::g1()\n"; }
    };

struct D: B {    };
int main () {
```

```
        D d, * pd;
        B b, * pb = &b, * pbd = &d;
        pd = dynamic_cast <D*> (pbd);
        pd -> g1();
        pd = dynamic_cast <D*> (pb);
        pd -> g1();
        return 0;
    }
```

Задача №8.

Дана заготовка КС-грамматики

$G = \langle \{A, B, S\}, \{a, b\}, P, S \rangle$, где $P = \{ A \to AA \mid a ;\ B \to b ; X \to \alpha \}$.

Найти такие X и α для G, чтобы одновременно выполнялись условия:

(а) G не является приведённой грамматикой;
(б) $L(G) \neq \emptyset$;
(в) грамматика G однозначна.

Экзаменационный билет состоит из двух теоретических вопросов и двух задач, например:

1. Состав и схема функционирования классической системы программирования.
2. Обработка исключений в C++. Преобразование типов в обработчиках исключений.
3. Задача.

Дана леволинейная грамматика G_{left} :

$S \to Sa \mid Aa$ $A \to Aa \mid b \mid a$	а) Применим ли метод рекурсивного спуска к данной грамматике? Обоснуйте ответ.	б) Постройте эквивалентную детерминированную праволинейную грамматику G_{right}

4. Задача.

Вычеркните неразрешимые вызовы функции f (...). Что напечатает получившаяся программа?

```
struct A {
        operator int () { return 1; }
    };

void f (double d, char c) { cout << "f (double, char) \n";}
void f (double d, int j) { cout << "f (double, int) \n";}
void f (A a, const char * p) { cout << "f (A, const char *) \n";}
void f (int i, const char * p) { cout << "f (int, const char *) \n";}

int main () {
        A a ;
        f (a, 0);
        f (a, 'a');
        f ('a', 0);
        return 0;
    }
```

Методические материалы для проведения процедур оценивания результатов обучения

Используется дифференцированная система оценки знаний и навыков. Оценка основывается на:

- результатах сдачи коллоквиума;
- контроле посещаемости занятий;
- результате сдачи экзамена.

При этом, в порядке исключения, учитывая высокую оценку работы в семестре, часть студентов может получить «отличные» и «хорошие» оценки за экзамен без сдачи устного экзамена.

ШКАЛА И КРИТЕРИИ ОЦЕНИВАНИЯ результатов обучения (РО) по дисциплине (модулю)

Оценка / РО и соответствующие виды оценочных средств	2	3	4	5
Знания *Коллоквиум, Экзамен*	Отсутствие знаний	Фрагментарные знания	Общие, но не структурированные знания	Сформированные систематические знания
Умения *Коллоквиум*	Отсутствие умений	В целом успешное, но не систематическое умение	В целом успешное, но содержащее отдельные пробелы умение (допускает неточности непринципиального характера)	Успешное и систематическое умение
Навыки (владения, опыт деятельности) *Экзамен*	Отсутствие навыков (владений, опыта)	Наличие отдельных навыков (наличие фрагментарного опыта)	В целом, сформированные навыки (владения), но используемые не в активной форме	Сформированные навыки (владения), применяемые при решении задач

Соответствие результатов обучения и компетенций, в развитии которых участвует дисциплина (модуль)

Результаты обучения	Компетенция, с частичным формированием которой связано достижение результата обучения
Знать: • основные понятия языка C++; • основные виды отношений между классами, ER-диаграммы; • основы параметрического полиморфизма, шаблоны функций и классов; • определение понятия "Системы программирования"; • основные требования к системам программирования; • основы теории формальных языков и грамматик; • классификацию формальных грамматик и языков по Хомскому; • основы теории трансляции; **Уметь:** • самостоятельно разрабатывать алгоритмы для решения задач системного и прикладного программирования.	ОПК-1

• применять на практике основные методы теории трансляции. **Владеть:** • навыками решения практических задач на языке С++; • навыками решение задач по теории трансляции; • навыками решения практических задач по теории формальных грамматик. **Иметь опыт:** • разработки компонентов программного обеспечения на основе использования объектно-ориентированной парадигмы программирования; • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	
Знать: • основные понятия языка С++; • понятие перегрузки функций и операций; • основные понятия наследования, единичное и множественное наследование; • понятие механизма виртуальных функций; • понятие об обработке исключений в С++; • основы параметрического полиморфизма, шаблоны функций и классов; • состав и схему функционирования классической системы программирования; • основы теории формальных языков и грамматик; • основы теории трансляции; • типы трансляторов и схемы их работы; • задачи и схемы работы лексического, синтаксического и семантического анализаторов. **Уметь:** • создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов; • самостоятельно разрабатывать алгоритмы для решения задач системного и прикладного программирования. • применять на практике основные методы теории трансляции. **Владеть:** • навыками решения практических задач на языке С++; • навыками решения практических задач по теории формальных грамматик. **Иметь опыт:** • разработки компонентов программного обеспечения на основе использования объектно-ориентированной парадигмы программирования;	ОПК-2
Знать: • основные понятия языка С++; • основные требования к системам программирования; • состав и схему функционирования классической системы программирования; • основы теории формальных языков и грамматик;	ОПК-3

• основы теории трансляции; • типы трансляторов и схемы их работы; **Уметь:** • самостоятельно разрабатывать алгоритмы для решения задач системного и прикладного программирования. • применять на практике основные методы теории трансляции. **Владеть:** • навыками решения практических задач на языке С++; • навыками решение задач по теории трансляции; • навыками решения практических задач по теории формальных грамматик. **Иметь опыт:** • разработки компонентов программного обеспечения на основе использования объектно-ориентированной парадигмы программирования; • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	
Знать: • основные понятия языка С++; • понятие перегрузки функций и операций; • основные понятия наследования, единичное и множественное наследование; • понятие механизма виртуальных функций; • основы параметрического полиморфизма, шаблоны функций и классов; • основные требования к системам программирования; • состав и схему функционирования классической системы программирования; • классификацию формальных грамматик и языков по Хомскому; • типы трансляторов и схемы их работы; • задачи и схемы работы лексического, синтаксического и семантического анализаторов. **Уметь:** • создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов; • применять метод преобразования недетерминированного конечного автомата к детерминированному; • применять метод рекурсивного спуска к КС- грамматикам; **Владеть:** • навыками решения практических задач на языке С++; • навыками решения практических задач по теории формальных грамматик. **Иметь опыт:** • разработки компонентов программного обеспечения на основе использования объектно-ориентированной парадигмы программирования; • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	ПК-6

Знать: • основные требования к системам программирования; • состав и схему функционирования классической системы программирования; • основы теории формальных языков и грамматик; • основы теории трансляции; • типы трансляторов и схемы их работы; • задачи и схемы работы лексического, синтаксического и семантического анализаторов. **Уметь:** • самостоятельно разрабатывать алгоритмы для решения задач системного и прикладного программирования. • применять на практике основные методы теории трансляции. **Владеть:** • навыками решение задач по теории трансляции; **Иметь опыт:** • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	ПК-7
Знать: • основные понятия языка С++; • основные понятия наследования, единичное и множественное наследование; • понятие механизма виртуальных функций; • понятие о динамической идентификации типов; • основы параметрического полиморфизма, шаблоны функций и классов; • состав и схему функционирования классической системы программирования; • основы теории формальных языков и грамматик; • основы теории трансляции; • типы трансляторов и схемы их работы; • задачи и схемы работы лексического, синтаксического и семантического анализаторов. **Уметь:** • самостоятельно разрабатывать алгоритмы для решения задач системного и прикладного программирования. • применять на практике основные методы теории трансляции. **Владеть:** • навыками решения практических задач на языке С++; • навыками решение задач по теории трансляции; • навыками решения практических задач по теории формальных грамматик. **Иметь опыт:** • разработки компонентов программного обеспечения на основе использования объектно-ориентированной парадигмы программирования; • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	ПК-9

8. Ресурсное обеспечение:

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, 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

3. 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

4. И. А. Волкова, А. А. Вылиток, Л.Е. Карпов. Сборник задач и упражнений по языку C++. Учебное пособие для студентовII курса.— М.: Издательский отдел факультета ВМК МГУ, 2013.
Электронная версия: http://cmcmsu.info/download/cpp.tasks.2013.pdf

5. 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

Дополнительная литература:

1. Страуструп Б. Язык программирования C++. Специальное изд./Пер. с англ. - М.: "Бином", 2015. *Электронная версия*:
https://codernet.ru/books/c_plus/bern_straustrup_yazyk_programmirovaniya_c_specialnoe_izdanie/

2.Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. Третье издание, М. ООО «И.Д.Вильямс», 2017.
Электронная версия: http://vmk.ugatu.ac.ru/book/buch/index.htm

3.Шилдт Г.. Самоучитель C++. – СПб, BHV, 2006.
Электронная версия: https://soft.sibnet.ru/soft/21999-samouciteli-c-3-e-izdanie/get/

4. А.В.Столяров. Введение в язык C++. Учебное пособие.- 4 изд. – М.МАКС Пресс, 2018.
Электронная версия: http://www.stolyarov.info/books/pdf/cppintro4.pdf

5. Страуструп Б. Программирование: принципы и практика использования C++.: Пер. с англ. – М. ООО «И.Д.Вильямс», 2016. *Электронная версия*:
http://i.uran.ru/webcab/system/files/bookspdf/programmirovanie-principy-i-praktika-s-ispolzovaniem-c/progr.pdf

6. Пол А. "Объектно-ориентированное программирование на Си++" – второе издание, М., Бином, 2001.*Электронная версия*:
https://codernet.ru/books/c_plus/obektno-orientirovannoe_programmirovanie_na_c/

7. .Д. Грис. Конструирование компиляторов для цифровых вычислительных машин. — М.: Мир, 1975.
Электронная версия: https://booksee.org/book/792044

8. .И. О. Одинцов. Профессиональное программирование. Системный подход. 2 изд. СПб.: БХВ-Петербург, 2004.
Электронная версия: http://static2.ozone.ru/multimedia/book_file/1000000748.pdf

Кроме того, студентам предлагается искать дополнительную информацию на сайтах, посвященных объектно-ориентированному программированию на языке C++, в частности на сайте комитета по стандартизации языка C++ (The C++ Standards Committee) http://www.open-std.org/JTC1/SC22/WG21/

Материально-техническое обеспечение:

Для проведения лекционных занятий необходима аудитория с партами и меловой доской, оборудованная проектором (или имеющая возможность подключения проектора) для показа презентаций и демонстрации решения типовых задач.

9. Язык преподавания.

Язык преподавания дисциплины — русский.

10. Преподаватель (преподаватели):

Доцент факультета ВМК МГУ Полякова И.Н.

11. Авторы программы:

Доцент факультета ВМК МГУ Полякова И.Н.