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

УТВЕРЖДАЮ
Декан факультета ВМК МГУ
_____ **/И.А.Соколов/**
«__» _____ **2023 г.**

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Наименование дисциплины (модуля):

Практикум на ЭВМ

Уровень высшего образования:

бакалавриат

Направление подготовки (специальность):

02.03.02 Фундаментальная информатика и информационные технологии

Направленность (профиль) ОПОП:

дисциплина относится к базовой части программы

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». Логически и содержательно-методически дисциплина поддерживает базовый лекционный курс «Системы программирования», читаемый в 4-ом семестре.

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

Компетенции выпускников, формируемые (полностью или частично) при реализации дисциплины (модуля):
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-2 - способность понимать, совершенствовать и применять современный математический аппарат, фундаментальные концепции и системные методологии, международные и профессиональные стандарты в области 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;

Планируемые результаты обучения по дисциплине (модулю):
Знать:

- основные понятия языка C++;
- понятие перегрузки функций и операций;
- основные понятия наследования, единичное и множественное наследование;
- понятие механизма виртуальных функций;
- понятие об обработке исключений в C++;
- основы параметрического полиморфизма, шаблоны функций и классов;
- основы теории формальных языков и грамматик;
- классификацию формальных грамматик и языков по Хомскому;
- понятие приведенных грамматик, алгоритм получения приведенной КС-грамматики;
- основы теории трансляции;

- алгоритм преобразования недетерминированного конечного автомата в детерминированный;
- метод рекурсивного спуска;
- понятие синтаксически управляемого перевода;
- ПОЛИЗ;
- задачи и схемы работы лексического и синтаксического анализаторов.

Уметь:

- создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов;
- разрабатывать алгоритмы для решения задач системного и прикладного программирования.
- применять алгоритм получения приведенной КС-грамматики;
- 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 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

Семинары по данной дисциплине проводятся как в формате традиционных занятий в аудиториях с использованием меловой доски, так и в виде практических занятий в компьютерном классе. На каждом занятии проводится обсуждение домашних заданий. Также все студенты имеют возможность задать преподавателю свои вопросы по изучаемой теме. Преподаватели, ведущие практические занятия, периодически проводят консультации по дисциплине.

5. Объем дисциплины (модуля) составляет 2 зачетные единицы, всего 72 часа,

в том числе 36 академических часа, отведенных на контактную работу обучающихся с преподавателем (семинары – 36 часа), 36 академических часов на самостоятельную работу обучающихся.

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

Данная дисциплина читается в поддержку основного лекционного курса «Системы программирования» базовой части цикла . Целью освоения дисциплины является получение навыков практического программирования на языке С++, а также умение решать задачи по теории формальных языков и грамматик, на которых базируются современные трансляторы языков программирования. Отдельной задачей ставится умение написать анализатор для

модельного языка программирования с применением объектно-ориентированного языка программирования С++ и элементов теории трансляции.

Дисциплина "Практикум на ЭВМ" состоит из двух частей:
1. Семинарские занятия в поддержку курса «Системы программирования».
2. Выполнение практических заданий по написанию и отладке программ на компьютерах.

Практические занятия посвящены совершенствованию навыков студентов в программировании на языке Си++. На практических занятиях студенты должны научиться самостоятельно писать программы на языке С++ для решения задач по изучаемым темам и, к концу семестра, самостоятельно выполнить задание практикума – написать лексический и синтаксический анализаторы модельного языка. Сдача задания подразумевает умение связно отвечать на вопросы по существу программы, а также способность вносить в свою программу дополнения и изменения по требованию преподавателя.

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. – указываются при необходимости)*
		Занятия лекционного типа*	Занятия семинарского типа*	**Всего**	
1.Обзор основных свойств и возможностей объектно-ориентированного программирования на языке Си++, рассмотренных в курсе “ООП” в 3 семестре. Работа с динамической памятью.	2	0	1	1	1
2. Статический полиморфизм в С++. Перегрузка функций и операторов. Перегрузка унарных операций с помощью функции-члена класса и с помощью функции-друга класса. Правила перегрузки бинарных и унарных операций в С++. Особенности перегрузки префиксных и постфиксных операций инкремента и декремента, операции ->, операции индексирования, операции присваивания, операции приведения к типу, операции ввода – вывода << и >>. Решение типовых задач.	4	0	2	2	2
3. Понятие наследования.	4	0	2	2	2

Единичное наследование. Наследование и замещение, правила видимости. Конструкторы и деструкторы производных классов: порядок вызова, передача параметров. Указатели на базовый и производный классы, преобразование указателей. Виртуальные функции. Абстрактные классы. Использование виртуальных деструкторов. Решение типовых задач.					
4. Практическое занятие в компьютерном классе по теме “Перегрузка операций”.	3	0	2	2	1
5. Практическое занятие в компьютерном классе по теме “Наследование”.	3	0	2	2	1
6. Множественное наследование в C++. Основные проблемы и способы их решения. Преобразование указателей. Виртуальные базовые классы. Неоднозначность из-за совпадающих имён в различных базовых классах.	4	0	2	2	2
7. Исключения в C++. Общая схема обработки исключений: try - throw – catch. Правило выбора обработчика исключения. Решение типовых задач.	4	0	2	2	2
8. Динамическая идентификация типа, преобразования типов. Средства RTTI.	2	0	1	1	1
9. Практическое занятие в компьютерном классе по теме “Исключения в C++”.	3	0	2	2	1
10. Параметрический полиморфизм. Программирование с помощью шаблонов. Шаблоны функций и классов .	2	0	1	1	1
11. Практическое занятие в компьютерном классе по теме “Шаблоны”.	2	0	1	1	1
12. Выдача задания практикума.	2	0	1	1	1
13. Текущий контроль успеваемости: контрольная работа № 1.	4	0	2	2	2
14. Основные понятия теории формальных грамматик. Эквивалентность грамматик. Типы	4	0	3	3	1

грамматик и языков по Хомскому. Дерево вывода. Неоднозначность грамматики. Неоднозначность языка. Приведенные грамматики. Преобразования грамматик. Решение типовых задач.					
15. Регулярные и автоматные грамматики. Леволинейные и праволинейные регулярные грамматики. Алгоритм получения леволинейной грамматики по праволинейной. Построение диаграммы состояний. Построение детерминированного конечного автомата по недетерминированному конечному автомату. Решение типовых задач.	4	0	3	3	1
16. Метод рекурсивного спуска. Подкласс грамматик, к которому применим метод. Критерий применимости метода рекурсивного спуска. Решение типовых задач.	3	0	2	2	1
17. Практическое занятие в компьютерном классе по теме “лексический и синтаксический анализаторы”.	4	0	2	2	2
18. Грамматики с действиями. Синтаксически управляемый перевод. Решение типовых задач.	3	0	2	2	1
19. ПОЛИЗ выражений, алгоритм интерпретации ПОЛИЗ. Алгоритм Дейкстры. ПОЛИЗ операторов языков программирования. Решение типовых задач.	3	0	2	2	1
20. Сдача задания практикума.	5	0	1	1	4
21. Текущий контроль успеваемости: контрольная работа № 2.	2	0	0	0	2
22. Промежуточная аттестация: зачет с оценкой.	5	0	0	0	5
Итого	72	0	36	36	36

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.

Контрольная работа № 1 «Язык Си++».

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

1. Что напечатает следующая программа?

```
class I {
        int i;
        public: I() : i(6) { cout << "owl" << endl; }
        I(int a) : i(a) { cout << "sheep " << i << endl; }
        I(const I & other) : i(other.i) { cout << "horse " << i << endl; }
        ~I() { cout << "wolf" << endl; }
        int Get() { return i; }
        void operator*=(const I & op) { i*=op.i; }
    };

void f (I x, I & y) {
                x *= 1; y *= x;
            }

int main() {
            I i1;
            I i2(3);
            i1 *= 7;
            f(i1, i2);
            cout << i1.Get() << ' ' << i2.Get()<< endl;
            return 0;
      }
```

2. Объект x принадлежит классу X, для которого определены конструктор преобразования типа int в класс X и функция преобразования класса X в тип int. .Описать прототипы двух перегруженных функций h из некоторой области видимости, для которых будут верны следующие обращения к ним:

```
h ("333");
h (x, 0);
h (1, "dd );
h (0,0);
h(x, "nn");
```

3. Найдите ошибки в программе и объясните, в чем они заключаются.

```
class table{
            int size;
            int priority;
            public:  table(int s=0,int p):size(s),priority(p){ }
                     virtual void print()=0;
            };

class stud_table:  public table{
                        char * name;
                        int gr;
                        public: void print(){cout<<"students table"<<endl;}
                        ~stud_table(){delete []name;}
          };

class asp_table: protected table{
                                char* thesis;
```

```
        };

int main(){
        table t;
        stud_table st;
        table * tp = &st;
        tp = new asp_table();
        stud_table * stp = &st;
        cout<<"Program"<<endl;
        return 0;
    }
```

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

```
struct X;
void f (X & x, int n);
int const P = 1;
int const Q = 1;
int const R = 1;

struct X {
    X() { try { f(*this, -1);
          cout << 1 << endl;
            }
         catch (X) { cout << 2 << endl; }
         catch (int) { cout << 3 << endl; }
        }
     X (X &) { cout << 4 << endl; }
    ~X () { cout << 5 << endl; }
    };

struct Y: X {
        Y () { f(*this, 1);
            cout << 6 << endl;
            }
        Y (Y &) { cout << 7 << endl; }
        ~Y () { cout << 8 << endl; }
         };
void f (X & x, int n) {
            try { if (n < 0) throw x;
                if (n > 0) throw 1;
                cout << 9 << endl;
                  }
            catch (int) { cout << 10 << endl; }
            catch (X& a) { cout << 11 << endl;
                           f(a, 0);
                           cout << 12 << endl;
                           throw;
                          }
          }

int main() {
```

```
    try { Y a; }
    catch (...) { cout << 13 << endl;
                  return 0;
                }
    cout << 14 << endl;
    return 0;
}
```

5. Для приведённой ниже программы описать функцию f (), которая, получая в качестве параметра ссылку на объект базового класса A, возвращает ссылку на объект производного класса C, полученную наиболее безопасным образом, а в случае невозможности приведения типов корректно завершает программу.

```
struct A {
        virtual void z () {}
      };

struct B: A { };

struct C: B {
            int x;
            C (int n = 3) { x = n; }
 };

C & f (A & ra);

int main () {
            C c, & pc = f (& c);
            cout << pc.x << endl;
            return 0;
            }
```

6. Для класса рациональных дробей с числителями и знаменателями некоторого интегрального типа template <class T> class fr { T n; T d; ... }; описать два варианта (методом класса и функцией-другом этого класса) реализации вне класса операций '*', выполняющей умножение одной рациональной дроби на другую.

Контрольная работа № 2 «Формальные грамматики и языки. Элементы теории трансляции».

1. а) Построить приведенную грамматику, эквивалентную заданной грамматике G:

S → aAb | CB
B→ bB
A→ aA | D
C→ cCc | B | c
D→ d

б) Каков тип получившейся грамматики?
в) Какой язык порождает грамматика?
г) Каков тип этого языка?

Для типов грамматики и языка указать максимально возможный номер по Хомскому.

2. Дана автоматная леволинейная грамматика G:

S → C⊥
A→ Aa | a
B→ Aa | b
C→ Ba
а) Детерминирован ли разбор по этой грамматике? Обосновать ответ.
б) Построить по грамматике конечный автомат и преобразовать его к детерминированному виду по алгоритму НКА→ДКА
в) Построить по получившемуся ДКА праволинейную автоматную грамматику.
г) Эквивалентны ли исходная и получившаяся автоматные грамматики? Обосновать ответ.
3. Дана КС-грамматика G:

S → A | B
A→ aA | ε
B→ bB | cC | ε
C→ aABC | c

а) Преобразовать грамматику G к неукорачивающей КС-грамматике с помощью алгоритма устранения пустых правых частей в КС грамматике (КС→НКС).
б) Применим ли метод рекурсивного спуска к исходной грамматике? Ответ обосновать.

4. Построить грамматику выражений, содержащих:
- целочисленные бинарные операции + и –
- односимвольные целые числа 1 и 3
Добавить _в грамматику действия (только вида cout << "символ";) по переводу заданных выражений в ПОЛИЗ во время анализа РС-методом. Приоритет операций + и – должен быть задан построенной грамматикой таким образом, чтобы ПОЛИЗ выражения:
3 -- 1 + 1
совпадал с ПОЛИЗом выражения со скобками:
(3 – (1 + 1))

Задание Практикума:
Реализовать на языке C++ лексический и синтаксический анализаторы для модельного языка.
Варианты языков – см. [5] из основного списка литературы.

Типовые задачи для практических занятий в компьютерном классе:
1. Для автоматизированной системы регистрации на курсы кройки и шитья описать классы Course и Student. Класс Course описывает один конкретный курс. В нем должно быть предусмотрено хранение информации об общем количестве слушателей этого курса, максимальное допустимом количестве слушателей для этого курса. У слушателя курсов (Student) должны быть определены фамилия, имя, номер паспорта, информация о курсе, на котором он учится. Предусмотреть функции добавления слушателя и исключения его с курсов. Один и тот же слушатель не может учиться на нескольких курсах. Написать функцию, не являющуюся членом указанных классов, которая для двух параметров Student проверяет, учатся ли они на одном курсе.

2. Описать шаблонную функцию max, которая возвращает значение максимального элемента для заданного массива целых чисел (int), длинных целых чисел (long), массива вещественных чисел (double) или массива строк (constchar*). При работе со строками, максимальной считать ту строку, которая имеет наибольшую длину.

3. Определить иерархию классов для описания сессии в университете. Базовый класс – event, содержит информацию о дате события, фамилию действующего лица и чисто виртуальную

функцию print_res, печатающую информацию о событии; классы-наследники – test (зачет), exam(экзамен). В них должны быть определены дополнительные характеристики событий и метод print_res. В функции main() определить сессию как массив указателей на события, проинициализировать элементы массива и распечатать информацию об экзаменах и зачетах.

4. Написать лексический анализатор на языке C++ по заданной автоматной грамматике:

Gleft=⟨ {a,b,⊥}, {S,A,B,C}, P, S⟩ с правилами P:

S⟶ C⊥
C⟶ Ab | Ba
A⟶ a | Ca
B⟶ b | Cb

5. Написать синтаксический анализатор на языке C++ методом рекурсивного спуска по заданной грамматике G:

S⟶ ABd
A⟶ a | cA
B⟶ bA

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

Необходимые (для получения зачета) знания и навыки:

- Определение типа по Хомскому заданной грамматики.
- Определение типа языка, порождаемого заданной грамматикой.
- Построение грамматики определенного типа, порождающей заданный язык.
- Определение языка, порождаемого заданной грамматикой.
- Построение приведенной грамматики.
- Устранение пустых правых частей заданной КС-грамматики.
- Построение ДС конечного автомата по заданной леволинейной грамматике.
- Построение анализатора на C++ по заданной ДС.
- Восстановление леволинейной регулярной грамматики, порождающей язык, распознаваемый конечным автоматом, заданным в виде ДС.
- Восстановление леволинейной регулярной грамматики по заданному тексту анализатора на C++.
- Построение ДС по праволинейной грамматике и построение праволинейной грамматики по заданной ДС.
- Преобразование праволинейной грамматики в эквивалентную леволинейную с помощью ДС.
- Преобразование НКА в эквивалентный ДКА с помощью соответствующего алгоритма.
- Определение и обоснование применимости метода рекурсивного спуска к заданной КС-грамматике.
- Построение (на C++) анализатора методом рекурсивного спуска для заданной КС-грамматики.
- Восстановление КС-грамматики по заданному анализатору, построенному методом рекурсивного спуска.
- Определение языка, порождаемого грамматикой с действиями.
- Дополнение заданной КС-грамматики действиями, позволяющими учесть дополнительные ограничения на цепочки определяемого ею языка.

- Вставка в заданную (или построенную) грамматику языка $L1$ действий по переводу цепочек этого языка в цепочки языка $L1$ по заданному закону соответствия между цепочками (т. е. реализовать
- формальный перевод).
- Определение языков $L1$ и $L1$ по заданной грамматике с действиями, реализующей формальный перевод языка $L1$ в язык $L1$.
- Запись в ПОЛИЗе заданного фрагмента программы на заданном языке.
- Восстановление текста на заданном языке заданного фрагмента программы в ПОЛИЗе.
- Определение, является ли данная запись ПОЛИЗом заданной конструкции.
- Вставка в грамматику, порождающую некоторую конструкцию языка программирования, действий, осуществляющих синтаксически управляемый перевод этой конструкции в ПОЛИЗ при анализе РС-методом.

‘+’ умение писать и анализировать программы на языке Си++ в рамках пройденной тематики.

Зачёт с оценкой.

Используется дифференцированная система оценки знаний и навыков. Зачёт проставляется по результатам освоения студентом дисциплины в семестре:
- контрольная работа №1,
- контрольная работа № 2,
- успешное выполнение практических заданий в компьютерном классе,
- сдача задания практикума.

Сдача программ подразумевает умение связно отвечать на вопросы по существу программы, а также способность вносить в свою программу дополнения и изменения по требованию преподавателя.

Для не отчитавшихся в течение семестра предусмотрена зачетная работа по типу контрольной №1, контрольной № 2, а также решение задач на компьютерах по изученным темам языка С++ и написание мини-анализаторов по регулярным и/или контекстно-свободным грамматикам (методом рекурсивного спуска).

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

Оценка / РО и соответствующие виды оценочных средств	2	3	4	5
Знания *Контрольные работы, Зачет*	Отсутствие знаний	Фрагментарные знания	Общие, но не структурированные знания	Сформированные систематические знания
Умения *Выполнение практических заданий, Сдача задания практикума, Зачет*	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) *Зачет*	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 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; • основы теории формальных языков и грамматик; • основы теории трансляции; **Уметь:** • создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов; • 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; • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	ОПК-2
Знать: • основные понятия языка С++; • понятие перегрузки функций и операций;	ОПК-3

• основные понятия наследования, единичное и множественное наследование; • понятие механизма виртуальных функций; • понятие об обработке исключений в С++; • основы параметрического полиморфизма, шаблоны функций и классов; • основы теории формальных языков и грамматик; • классификацию формальных грамматик и языков по Хомскому; • понятие приведенных грамматик, алгоритм получения приведенной КС-грамматики; • основы теории трансляции; • алгоритм преобразования недетерминированного конечного автомата в детерминированный; • метод рекурсивного спуска; • понятие синтаксически управляемого перевода; • ПОЛИЗ; • задачи и схемы работы лексического и синтаксического анализаторов. **Уметь:** • создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов; • разрабатывать алгоритмы для решения задач системного и прикладного программирования. • применять метод преобразования недетерминированного конечного автомата к детерминированному; • применять алгоритм получения приведенной КС-грамматики; • применять метод рекурсивного спуска к КС-грамматикам; • 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; • решения практических задач, связанных с использованием языка программирования Си++ и элементов теории трансляции.	

Знать: • основные понятия языка С++; • основы теории формальных языков и грамматик; • основы теории трансляции; **Уметь:** • создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов; • 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;	ПК-2
Знать: • основные понятия языка С++; • основы теории формальных языков и грамматик; • классификацию формальных грамматик и языков по Хомскому; • основы теории трансляции; • алгоритм преобразования недетерминированного конечного автомата в детерминированный; • метод рекурсивного спуска; • задачи и схемы работы лексического и синтаксического анализаторов. **Уметь:** • создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на языке С++, оценивать сложность полученных алгоритмов; • разрабатывать алгоритмы для решения задач системного и прикладного программирования. • применять на практике основные методы теории трансляции. **Владеть:** • навыками решения практических задач на языке С++;	ПК-7

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.	

8. Ресурсное обеспечение:

Основная литература:

1 .И. А. Волкова, А. В. Иванов, Л. Е. Карпов. Основы объектно-ориентированного программирования. Язык программирования С++. 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. А.В.Столяров. Введение в язык С++. Учебное пособие.- 4 изд. – М.МАКС Пресс, 2018.
Электронная версия: http://www.stolyarov.info/books/pdf/cppintro4.pdf

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

5. РуденкоТ.В. Интерпретатор модельного языка программирования.
Электронная версия: http://cmcmsu.info/download/model.lang.practical.task.pdf

6. Ю.С. Корухова. Сборник задач и упражнений по языку С++. 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 С++. Специальное изд./Пер. с англ. - М.: "Бином", 2015.
Электронная версия:
https://codernet.ru/books/c_plus/bern_straustrup_yazyk_programmirovaniya_c_specialnoe_izdanie/

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

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

4. МейерсС. Эффективное использование С++. 50 рекомендаций по улучшению ваших программ и проектов. Питер, ДМК Пресс, Москва, 2006
Электронная версия: https://booksee.org/book/751912

5. Страуструп Б. Программирование: принципы и практика использования С++.: 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

Материально-техническое обеспечение:

Для проведения аудиторных семинарских занятий необходима аудитория с партами и меловой доской.
Для проведения практических занятий необходимо наличие компьютерного класса с возможностью работы с компилятором языка C++.

9. Язык преподавания.

Язык преподавания дисциплины — русский.

10. Преподаватель (преподаватели):

Доцент кафедры алгоритмических языков факультета ВМК МГУ Полякова И.Н.
Ассистент кафедры алгоритмических языков факультета ВМК МГУ Кузина Л.Н.

11. Автор программы:

Доцент кафедры алгоритмических языков факультета ВМК МГУ Полякова И.Н.