Государственное бюджетное образовательное учреждение высшего образования Московской области

УНИВЕРСИТЕТ «ДУБНА»
(государственный университет «Дубна»)

Факультет естественных и инженерных наук

Кафедра «Физико-технические системы»

УТВЕРЖДАЮ
Проректор по учебно-методической работе
_____А.С. Деникин
«23» 03 2016 г.

Программа дисциплины

Языки программирования

Направление подготовки
13.03.02 «Электроэнергетика и электротехника»

Профиль подготовки
«Нетрадиционные и возобновляемые источники энергии»

Квалификация выпускника
Бакалавр

Форма обучения
очная

Дубна, 2016 г.

Преподаватель (преподаватели):

Никитин В.К., ст. преподаватель кафедры «Физико-технические системы» _____

Фамилия И.О., должность, ученая степень, ученое звание, кафедра; подпись

Рабочая программа разработана в соответствии с требованиями ФГОС ВО с учетом рекомендаций примерной образовательной программы по направлению подготовки (специальности) высшего образования

13.03.02 «Электроэнергетика и электротехника»

(код и наименование направления подготовки (специальности))

Программа рассмотрена на заседании кафедры «Физико-технические системы»

Протокол заседания № 3 от «19» 05 2016 г.

Заведующий кафедрой _____ /Малахов А.И./

СОГЛАСОВАНО

И.о./Декан факультета _____ /Савватеева О.А./

Рецензент: _____ (Кобышев М.Н.) нач. бригады АО. ГосМКБ. Радуга" им. А.Я. Березняка"

(ученая степень, ученое звание, Ф.И.О., место работы, должность)

Акционерное общество "Государственное машиностроительное конструкторское бюро "Радуга" имени А.Я. Березняка" Российская Федерация Московская область г. Дубна ОТДЕЛ КАДРОВ

Содержание

1. Цели и задачи дисциплины.

Цель дисциплины: изучение методов программирования для овладения знаниями в области технологии программирования; подготовка к осознанному использованию, как языков программирования, так и методов программирования.

Основные задачи курса программирования на основе структурного и объектно-ориентированного подхода:

- знакомство с методами структурного и объектно-ориентированного программирования как наиболее распространенными и эффективными методами разработки программных продуктов;
- обучение разработке алгоритмов на основе структурного и объектно-ориентированного подхода;
- закрепление навыков алгоритмизации и программирования на основе изучения языка программирования C++;
- знакомство с основными структурами данных и типовыми методами обработки этих структур;

Концепция дисциплины основана на том, что она имеет общеобразовательный и в определенной степени мировоззренческий характер и предназначена для формирования специалиста с широким научным кругозором.

2. Место курса в профессиональной подготовке студентов

Изучение дисциплины Информатика (раздел Языки программирования) базируется на знании математических дисциплин и общего курса информатики.

Материалы курса является важной составляющей при работе над курсовыми работами в ряде специальных курсов и при подготовке дипломного проекта.

2.1. Формы работы студентов

В ходе изучения дисциплины предусмотрены практические занятия и выполнение домашних работ. Отдельные темы теоретического курса прорабатываются студентами самостоятельно в соответствии с планом самостоятельной работы и конкретными заданиями преподавателя с учетом индивидуальных особенностей студентов.

Самостоятельная работа студентов, предусмотренная учебным планом, выполняется в ходе семестра в форме подготовки к семинарским занятиям и выполнения домашних работ.

Перечень обязательных видов работы студента:

- посещение лекционных занятий;
- ответы на теоретические вопросы на семинаре;
- решение практических задач и заданий на семинаре;
- выполнение контрольных работ;
- выполнение домашних работ.

2.2. Форма текущего и итогового контроля

Текущий контроль заключается в проверке домашних заданий. Этапный контроль проводится с целью определения качества усвоения пройденного лекционного материала. Наиболее эффективным является его проведение в форме практических заданий – по контрольным вопросам, тестам, и т.п., в виде сдачи всеми без исключения студентами задач во время проведения практических занятий.

В ходе изучения дисциплины студенты выполняют контрольные работы, сдают экзамен по теоретической и по практической части.

3.Требования к уровню освоения содержания дисциплины.

Изучение дисциплины «Языки программирования» направлено на формирование у студентов следующих компетенций:

- способность осуществлять поиск, хранение, обработку и анализ информации из различных источников и баз данных, представлять ее в требуемом формате с использованием информационных, компьютерных и сетевых технологий (ОПК-1);

- способность применять соответствующий физико-математический аппарат, методы и анализы моделирования, теоретического и экспериментального исследования при решении профессиональных задач (ОПК-2)

В результате изучения дисциплины Информатика (раздел Языки программирования) студент должен:

иметь представление:

- о конструировании алгоритмов,
- о методах структурного и модульного программирования,
- об абстракциях основных структур данных (списки, множества и т.п.) и методах их обработки и способах реализации,
- о методах и технологиях программирования;

уметь:

- разрабатывать алгоритмы,
- реализовывать алгоритмы на языке программирования высокого уровня,
- описывать основные структуры данных,
- реализовывать методы обработки данных,
- делить степень усвоения студентом уче*приобрести навыки:*

приобрести навыки:

- структурного программирования,
- алгоритмизации,
- работы в среде программирования (составление, отладка и тестирование программ; разработка и использование интерфейсных объектов).

владеть, иметь опыт:

- разработки алгоритмов,
- описания структур данных,
- описания основных базовых конструкций,
- программирования на языке высокого уровня,
- работы в различных средах программирования.

4. Содержание и структура дисциплины.

4.1. Введение в программирование

1.1. Методологии программирования. Программирование как раздел информатики. История и эволюция информатики.

1.2. Алгоритмические структуры. Этапы решения задач на ЭВМ. Понятие алгоритма. Способы записи алгоритмов. Принципы структурного программирования. Язык программирования. Классификация языков программирования.

4.2. Язык программирования С++

2.1. Общая характеристика языка С++. Структуры данных. Определение констант. Описание переменных. Стандартные типы данных. Целые типы. Символьный и булевский типы данных. Эквивалентность и совместимость типов. Типы, определяемые программистом: перечисляемый, интервальный. Тип дата-время.

2.2. Основные операторы языка. Перечень операторов C++. Оператор присваивания. Операторы (процедуры) ввода-вывода. Управление выводом данных в консольном режиме (простейшее форматирование). Условный оператор. Логические выражения. Оператор множественного ветвления. Операторы цикла: с предусловием, с постусловием, с параметром.

2.3. Структурированные типы языка программирования высокого уровня. Массивы. Примеры задач с численными, символьными, булевскими массивами. Строковый тип данных. Записи. Оператор присоединения. Операции отношения.

2.4. Файлы. Понятие логического и физического файлов. Файловые типы. Общие процедуры для работы с файлами. Типизированные файлы. Текстовые файлы. Нетипизированные файлы и процедуры ввода-вывода. Прямой и последовательный доступ к компонентам файлов.

2.5. Алгоритмы поиска и сортировки. Простой и бинарный поиск. Сортировки: выбором, обменом, вставкой. Анализ сложности алгоритмов на примере сортировок.

4.3. Модульное программирование. Программирование абстрактных типов данных

3.1. Процедуры и функции. Модули. Подпрограммы. Формальные параметры. Параметры-значения, параметры-переменные, параметры-константы. Локальные и глобальные идентификаторы подпрограмм. Процедуры и функции. Рекурсия. Внешние подпрограммы. Модули. Общая структура модуля. Подпрограммы в модулях. Компиляция и использование модулей.

3.2. Организация динамических структур данных (абстрактных типов данных). Динамические структуры. Динамическое распределение памяти. Виды списков. Примеры использования списков. Организация динамических структур данных: стек, очередь, двоичное дерево поиска.

4.4. Объектно-ориентированное программирование

4.1. Введение в объектно-ориентированное программирование. Введение в объектно-ориентированное программирование (ООП) и проектирование. Инкапсуляция, наследование, полиморфизм. Примеры задач.

4.2. Реализация абстракций данных методами объектно-ориентированного программирования. Математические объекты: рациональные и комплексные числа, вектора, матрицы. Библиотеки объектов.

4.3. Объектно-событийное и объектно-ориентированное программирование. Идеология программирования под Windows. Событие и сообщение. Виды событий. События от мыши и клавиатуры. Программирование управления событиями. Обработка исключительных событий. Основы визуального программирования. Компонент. Иерархия компонентов.

4.5. Структура дисциплины и виды учебной работы[1]

Вид учебной работы	Всего часов	Семестры	
		5	
Общая трудоёмкость	72	72	
Аудиторные занятия:	34	34	
Лекции			

[1] Объем часов на каждый вид учебной работы должен обязательно быть согласован с учебным планом по данной специальности.

Практические занятия (ПЗ)	34	34	
Семинары			
Лабораторные работы (ЛР)			
Самостоятельная работа:	38	38	
Контрольные работы			
Реферат			
Курсовая работа			
Промежуточная аттестация		зачет	

4.6. Практические занятия (семинары).

№ п.п.	№ раздела дисциплины	Наименование практических занятий (семинаров)
1.	2	Алгебраические и логические выражения, правила их записи.
2.	2	Присваивание. Совместимость по присваиванию.
3.	2	Ввод и вывод данных в консольном режиме.
4.	2	Условный оператор. Оператор выбора.
5.	2	Операторы цикла (циклы с пред- и постусловием, цикл с параметром).
6.	2	Характеристики структурированных типов данных. Массивы. Линейные и двумерные массивы. Строки.
7.	2	Типизированные файлы. Организация файлов записей. Нетипизированные файлы.
8.	2	Текстовые файлы. Прямой доступ к компонентам файлов. Сортировка файлов.
9.	3	Процедуры. Разработка и вызов.
10.	3	Функции. Разработка и вызов.
11.	3	Разработка программ на основе структурного подхода.
12.	3	Внешние подпрограммы.
13.	3	Рекурсивные подпрограммы.
14.	3	Модули. Структура и разработка. Стандартные модули.
15.	3	Динамически распределяемая память и ее использование при работе со стандартными типами данных.
16.	3	Однонаправленные списки. Двунаправленные списки.
17.	4	Основные понятия ООП.
18.	4	Разработка программ на основе ООП.
19.	4	Наследование и полиморфизм в ООП.
20.	4	Абстрактные типы и структуры данных.
21.	4	Классы, объекты, поля, методы.
22.	4	Конструкторы и деструкторы.
23.	4	Свойства и методы объектов.
24.	4	Раннее связывание и позднее связывание.
25.	4	Математические объекты: рациональные и комплексные числа, вектора, матрицы.
26.	4	Контрольная работа.

5. Материально-техническое обеспечение дисциплины

1. Компьютерные презентации по материалам лекций
2. Учебная аудитория, оборудованная персональными компьютерами с набором программного обеспечения: системы программирования Borland C++ Builder 6.

6. Формы контроля и оценочные средства для текущего контроля успеваемости и промежуточной аттестации по итогам освоения дисциплины[2]

Темы контрольных работ и варианты заданий для них:

Контрольная работа по пройденному материалу. Задания контрольной работы приведены в разделе «Материалы, устанавливающие содержание и порядок проведения текущего контроля успеваемости и промежуточной аттестации».

Вопросы, выносимые на экзамен (8 семестр):

1. Этапы решения задач с использованием ЭВМ.
2. Понятие алгоритма. Подходы к определению алгоритма. Свойства алгоритма. Способы записи алгоритма.
3. Информатика (раздел Языки программирования). Алгоритмические языки (алфавит, синтаксис, семантика). Способы описания синтаксиса (язык металингвистических формул, синтаксические диаграммы).
4. Система программирования C++.
5. Структура программы, элементы языка C++ (алфавит). Понятие типа данных.
6. Операции (арифметические, логические) на типах. Стандартные функции. Выражения.
7. Процедуры консольного ввода и вывода, управление вводом-выводом. Оператор присваивания. Совместимость по присваиванию.
8. Условный оператор. Оператор множественного ветвления (выбора).
9. Циклы в C++: с предусловием, с постусловием. Связь с другими циклами.
10. Циклы в C++: с параметром. Связь с другими циклами.
11. Структурированные типы данных. Линейные массивы. Примеры задач.
12. Структурированные типы данных. Двумерные массивы. Примеры задач.
13. Сортировка массивов. Метод выбора. Двоичный поиск в массиве.
14. Сортировка массивов. Метод обмена.
15. Сортировка массивов. Метод вставок.
16. Подпрограммы в C++. Основные способы передачи параметров в подпрограмму, их сравнение.
17. Подпрограммы в C++. Область видимости. Локальные и глобальные идентификаторы.
18. Процедуры. Организация и вызов. Примеры.
19. Функции. Организация и вызов. Примеры.
20. Простые типы данных в C++.
21. Структурированные типы данных. Строковый тип данных в C++: основные процедуры и функции, примеры.
22. Рекурсия. Механизм рекурсии. Примеры.
23. Сортировка массивов. Метод быстрой сортировки.
24. Комбинированный тип данных (записи). Оператор присоединения. Записи с вариантами. Программирование типовых алгоритмов обработки записей.
25. Файловые типы в C++. Общие процедуры для работы с файлами. Компонентные (типизированные) файлы.
26. Текстовые файлы. Текст-ориентированные процедуры и функции. Типовые задачи.

---

[2] Описывать следует только те формы контроля, которые предусмотрены программой дисциплины

27. Прямой и последовательный доступ к компонентам файла. Процедуры и функции, ориентированные на прямой доступ к компонентам файла.
28. Поиск в типизированных файлах. Сортировка файлов (на примере одного из методов).
29. Типизированные файлы. Файлы записей. Типовые алгоритмы обработки.
30. Статическая и динамически распределяемая память. Пример использования указателей.
31. Динамические структуры данных. Однонаправленный список. Процедуры обработки списка.
32. Динамические структуры данных. Двунаправленный список. Процедуры обработки списка.
33. Динамические структуры данных. Кольцевой список (однонаправленный или двунаправленный). Процедуры обработки списка.
34. Стек. Процедуры обработки.
35. Очередь. Процедуры обработки.
36. Двоичное дерево. Добавление в дерево и поиск в дереве.
37. Двоичное дерево. Удаление элемента из дерева.
38. Модуль. Общая структура модуля. Компиляция и подключение модуля.
39. Объектно-ориентированное программирование.
40. Пример реализации задачи на ООП в Builder C++.

7. Учебно-методические материалы

1. Учебно-методические материалы для студентов:

Методические рекомендации по организации самостоятельной работы студентов:

Рабочей программой настоящей дисциплины предусмотрена самостоятельная работа студентов. Самостоятельная работа проводится с целью углубления знаний по дисциплине и предусматривает:

- чтение студентами рекомендованной литературы и усвоение теоретического материала дисциплины;
- подготовку к практическим занятиям;
- работу с Интернет-источниками;
- подготовку к сдаче коллоквиумов, выполнению тестовых заданий и сдаче зачетов и экзаменов.

Планирование времени на самостоятельную работу, необходимого на изучение настоящей дисциплины, студентам лучше всего осуществлять на весь семестр, предусматривая при этом регулярное повторение пройденного материала. Материал, законспектированный на лекциях, необходимо регулярно дополнять сведениями из литературных источников, представленных в рабочей настоящей программе дисциплины. По каждой из тем для самостоятельного изучения, приведенных в рабочей программе дисциплины следует сначала прочитать рекомендованную литературу и при необходимости составить краткий конспект основных положений, терминов, сведений, требующих запоминания и являющихся основополагающими в этой теме и для освоения последующих разделов курса.

Для расширения знаний по дисциплине рекомендуется использовать Интернет-ресурсы: проводить поиск в различных системах, сайтов и обучающих программ, рекомендованных преподавателем на лекционных занятиях.

Правила выполнения и оформления домашних работ:

В процессе самостоятельного изучения настоящего курса каждый студент должен выполнить домашние работы с защитой у преподавателя. Эти работы позволяют определить степень усвоения студентом учебного материала и предусматривают:

1. Самостоятельную работу с учебной литературой.
2. Решение задач на закрепление материала по различным разделам курса.

При выполнении работ студент должен придерживаться следующих требований:

1. Работу рекомендуется выполнять в виде отдельного проекта в среде C++ Builder 6, с указанием номера группы, Ф.И.О. студента.
2. Программа должна быть выполнена аккуратно, расположение операторов и команд не должен вызывать затруднений при прочтении программы.
3. При оформлении программы необходимо писать краткие комментарии около каждого блока или подпрограммы.

Преподаватель оценивает работу по рейтинговой системе. Баллы могут быть начислены и за незавершенную работу. Если полученных баллов недостаточно для получения удовлетворительной оценки, то работа возвращается студенту для исправления и доработки, после чего снова должна быть представлена на проверку.

Студенты, не выполнившие домашние, проверочные и лабораторные работы, не допускаются к зачетной и экзаменационной сессии.

2. Методические рекомендации для преподавателей:

Одной из задач преподавателей, ведущих занятия по настоящей дисциплине является выработка у студентов осознания важности, необходимости и полезности знания дисциплины для дальнейшей работы их инженерами, специалистами. Методическая модель преподавания дисциплины основана на применении активных методов обучения.

Принципами организации учебного процесса являются:

- активное участие студентов в учебном процессе;
- проведение практических занятий, определяющих приобретение навыков решения проблемы;
- приведение примеров применения изучаемого теоретического материала к реальным практическим ситуациям.

Используемые методы преподавания: лекционные занятия с использованием мультимедиа технологий; индивидуальные и групповые задания при проведении практических и лабораторных занятий.

Для более глубокого изучения предмета преподаватель предоставляет студентам информацию о возможности использования по разделам дисциплины Интернет-ресурсов, кафедральной библиотеки.

Содержание занятий определяется календарным планом. Пакет заданий для самостоятельной работы следует выдавать в начале семестра, определив предельные сроки их выполнения и сдачи. Организуя самостоятельную работу, необходимо постоянно обучать студентов методам такой работы.

При наличии академических задолженностей по практическим занятиям, связанных с их пропусками преподаватель должен выдать задание студенту в виде задач по пропущенной теме занятия.

Для контроля знаний студентов по данной дисциплине необходимо проводить текущий и промежуточный контроль.

Текущий контроль проводится с целью определения качества усвоения лекционного материала. Наиболее эффективным является его проведение в письменной форме – по контрольным вопросам, тестам и т.п. Контроль проводится в виде сдачи всеми без исключения студентами контрольных заданий – задач во время проведения практических занятий. В материалы письменных опросов студентов включаются и темы, предложенные им для самостоятельной подготовки. В течение работы над освоением дисциплины студенты, руководствуясь календарным планом, выполняют контрольных работы, проводятся коллоквиумы, выполняется курсовая работа.

Промежуточный контроль по курсу. Для контроля усвоения данной дисциплины учебным планом предусмотрен экзамен. На экзамене студентам предлагается решить 1 практическую задачу и ответить на 2 вопроса по материалам учебной дисциплины. Ответы на поставленные вопросы даются в устном виде. Оценка по экзамену является итоговой по курсу и проставляется в приложении к диплому.

8. Материалы, устанавливающие содержание и порядок проведения текущего контроля успеваемости и промежуточной аттестации

1. Варианты заданий для контрольной работы и проведения экзамена:

Задание 1. Объяснить, в чем заключается синтаксическая ошибка (или ошибки) в приведенной программе. Написать тот вариант программы, который, по Вашему мнению, будет правильным.

```
void My_Error()
{
     int X = 10, Y, Sum;
     double v = exp(150);
     Y = v;
     Sum = X + Y;
     cout >> "Сумма равна "  << Sum
}
```

Задание 2. Составить программу идентификации треугольника по сторонам a, b, c. Определяемое свойство: является прямоугольным или не является прямоугольным (удобно использовать сравнение квадрата одной из сторон с суммой квадратов двух других сторон).

Задание 3. Решить задачу с использованием оператора выбора: по введенному числу грибов k напечатать фразу «Мы нашли в лесу k грибов» и согласовать при этом окончание слова «гриб» с числом k. (Количество грибов может быть любым целым числом: 1, 3, 34, 127 и т.п. Окончание фразы определяется значением последней цифры.)

Задание 4. Сформулировать условие задачи, которая решается в данной программе:

```
#include <stdio.h>
#include <conio.h>
main()
{
     int N,n;
     double x,s;
     printf("Введите N: "); scanf("%d", &N);
     printf("\n");
     printf("Введите x: "); scanf("%lf", &x);
     printf("\n");
     s = 0;
     for(n=1;n<=N;n++)
     {
          s += x/(3*n+2);
          x *= x;
     }
     printf("S = %e",s);
     getch();
 }
```

Задание 5. Решить задачу с использованием цикла: найти k-е простое число в арифметической прогрессии 11, 21, 31, 41, 51, 61, …

Задание 6. Решить задачу: подсчитать количество элементов массива, каждый из которых не меньше всех своих предшественников (т.е. элементов с меньшими индексами). Например, для массива –2, 3, 0, 13, 13, 4, –3 ответ будет 4.

Задание 7. Решить задачу с использованием подпрограммы: вывести наибольшую из первых цифр трех заданных чисел. Например, если a = 25, b = 730, c = 1995, то надо вывести цифру 7.

Задание 8. В данной последовательности a1, a2, …, an определить максимальное число среди элементов с номерами, кратными числу k. Например, для последовательности –1, 0, 12, -77, 22, –6, 70, 11, 3 и k = 3 получаем ответ 12.

Задание 9. Вычислить $\sqrt{2+\sqrt{2+\text{L}\sqrt{2}}}$ для n слагаемых.

Задание 10. Заполнить файл целыми числами из отрезка [-1000; 1000] с помощью датчика случайных чисел. Переписать в один из новых файлов те из компонент исходного файла, модуль которых является простым числом, в другой — все остальные.

Задание 11. Удалить из файла, содержащего целые числа, неположительные компоненты. Дополнительных файлов и массивов не использовать.

2. Пример экзаменационного билета:

«Университет «Дубна»
Факультет естественных и инженерных наук

Кафедра «Физико-технические системы»
Дисциплина: Языки программирования
Курс 4, семестр 8

Экзаменационный билет №1

1. Циклы в С++: с предусловием, с постусловием. Связь с другими циклами.
2. Файловые типы в С++. Общие процедуры для работы с файлами. Компонентные (типизированные) файлы.
3. Задача: В заданной строке заменить указанный символ С на символ С1. Подсчитать число замен.

и.о. зав. кафедрой Малахов А.И.

3. Вопросы для самоконтроля:

1. Этапы решения задач с использованием ЭВМ.
2. Понятие алгоритма. Подходы к определению алгоритма. Свойства алгоритма. Способы записи алгоритма.
3. Информатика (раздел Языки программирования). Алгоритмические языки (алфавит, синтаксис, семантика). Способы описания синтаксиса (язык металингвистических формул, синтаксические диаграммы).
4. Система программирования С++.
5. Структура программы, элементы языка С++ (алфавит). Понятие типа данных.
6. Операции (арифметические, логические) на типах. Стандартные функции. Выражения.
7. Процедуры консольного ввода и вывода, управление вводом-выводом. Оператор присваивания. Совместимость по присваиванию.
8. Условный оператор. Оператор множественного ветвления (выбора).
9. Циклы в С++: с предусловием, с постусловием. Связь с другими циклами.
10. Циклы в С++: с параметром. Связь с другими циклами.
11. Структурированные типы данных. Линейные массивы. Примеры задач.
12. Структурированные типы данных. Двумерные массивы. Примеры задач.
13. Сортировка массивов. Метод выбора. Двоичный поиск в массиве.
14. Сортировка массивов. Метод обмена.
15. Сортировка массивов. Метод вставок.
16. Подпрограммы в С++. Основные способы передачи параметров в подпрограмму, их сравнение.
17. Подпрограммы в С++. Область видимости. Локальные и глобальные идентификаторы.
18. Процедуры. Организация и вызов. Примеры.
19. Функции. Организация и вызов. Примеры.
20. Простые типы данных в С++.
21. Структурированные типы данных. Строковый тип данных в С++: основные процедуры и функции, примеры.
22. Рекурсия. Механизм рекурсии. Примеры.
23. Сортировка массивов. Метод быстрой сортировки.
24. Комбинированный тип данных (записи). Оператор присоединения. Записи с вариантами. Программирование типовых алгоритмов обработки записей.

25. Файловые типы в C++. Общие процедуры для работы с файлами. Компонентные (типизированные) файлы.
26. Текстовые файлы. Текст-ориентированные процедуры и функции. Типовые задачи.
27. Прямой и последовательный доступ к компонентам файла. Процедуры и функции, ориентированные на прямой доступ к компонентам файла.
28. Поиск в типизированных файлах. Сортировка файлов (на примере одного из методов).
29. Типизированные файлы. Файлы записей. Типовые алгоритмы обработки.
30. Статическая и динамически распределяемая память. Пример использования указателей.
31. Динамические структуры данных. Однонаправленный список. Процедуры обработки списка.
32. Динамические структуры данных. Двунаправленный список. Процедуры обработки списка.
33. Динамические структуры данных. Кольцевой список (однонаправленный или двунаправленный). Процедуры обработки списка.
34. Стек. Процедуры обработки.
35. Очередь. Процедуры обработки.
36. Двоичное дерево. Добавление в дерево и поиск в дереве.
37. Двоичное дерево. Удаление элемента из дерева.
38. Модуль. Общая структура модуля. Компиляция и подключение модуля.
39. Объектно-ориентированное программирование.
40. Пример реализации задачи на ООП в Builder C++.

9. Учебно-методическое обеспечение дисциплины

7.1. Основная литература[3]

1. **Павловская Т.А.** C/C++. Программирование на языке высокого уровня: Для магистров и бакалавров: Учебник для вузов / Павловская Татьяна Александровна; Рец. Г.И.Ревунков и др. - СПб.: Питер, 2011. - 464с.
2. **Архангельский А.Я.** Язык C++ в C++ Builder: Справочное и методическое пособие / Архангельский А.Я. - М.: Бином-Пресс, 2008. - 944с.
3.

7.2 Дополнительная литература

1. **Культин Н.Б.** Самоучитель C++ Builder в задачах и примерах / Культин Никита Борисович. - СПб.: БХВ-Петербург, 2007. - 336с.
2. **Страуструп Б.** Язык программирования C++ / Страуструп Бьерн; Пер.с англ. С.Анисимова, М.Кононова; Под ред. Ф.Андреева, А.Ушакова. - 3-е изд. - СПб.: Невский Диалект; М.: БИНОМ, 1999. - 991с.
3. **Borland C++ Builder 6**: Руководство разработчика / Холингворт Джаррод, Сворт Боб, Кэшмэн Марк, Густавсон Поль; Пер.с англ.В.Н.Заики и др.; Под ред. И.В.Красикова. - М.: Вильямс, 2004. - 976с.
4. **Задачи по программированию** / Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н., Селюн М.И. - М.: Наука, 1988. - 224с.

7.3. Программное обеспечение современных информационно-коммуникационных технологий

[3] Список основной литературы должен включать только источники, имеющиеся в наличии в библиотечной системе университета и удовлетворяющие предъявляемым требованиям. Необходимо согласование с руководителем библиотечной системы.

8. Материально-техническое обеспечение дисциплины

1. Компьютерный класс
2. Оборудование Инжинирингового центра Университета «Дубна»