АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ НОВЫЙ УНИВЕРСИТЕТ»
(АНО ВО «РосНОУ»)

Институт информационных систем и инженерно-компьютерных технологий

Кафедра 11 информационных технологий и естественнонаучных дисциплин

Рабочая программа учебной дисциплины

Основы программирования

Код и направление подготовки **13.03.02 Электроэнергетика и электротехника**

Заведующий кафедрой «11 Информационных технологий и
естественнонаучных дисциплин»
_____ к.т.н. Матюнина О.Е.

Москва

1. НАИМЕНОВАНИЕ И ЦЕЛЬ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Учебная дисциплина «Программирование» предназначена для обучающихся по направлению подготовки бакалавров «Электроэнергетика и электротехника» по профилю Электрооборудование и электрохозяйство предприятий, организаций и учреждений в соответствии с Федеральным государственным образовательным стандартом высшего образования по направлению 13.03.02 Электроэнергетика и электротехника (уровень бакалавриата), утвержденным приказом Министерства образования и науки РФ от 28.02.2018 N 144 (ФГОС ВО 3++).

Изучение учебной дисциплины направлено на подготовку обучающихся к осуществлению деятельности по обслуживанию оборудования подстанций электрических сетей, выполнению обобщенной трудовой функции по управлению деятельностью по техническому обслуживанию и ремонту оборудования подстанций, планированию и контролю деятельности по техническому обслуживанию и ремонту оборудования подстанций, организации работы подчиненного персонала, определенных профессиональным стандартом «Работник по обслуживанию оборудования подстанций электрических сетей», утвержденного приказом Министерства труда и социальной защиты Российской Федерации от 29.12.2015 г. № 1177н (Регистрационный номер №828).

Цель курса «Основы программирования» – развитие у студентов навыков программирования, способностей к самостоятельной творческой работе, умения применять методы программирования для решения различных задач прикладных дисциплин.

Задачами курса являются: обучение студентов основам программирования, включая постановку задачи, выбор метода решения задачи, создание или выбор алгоритма, реализацию алгоритма на языке программирования, отладку и тестирование программы.

Формирование представления о технологиях программирования и проектирования программных продуктов и применения их для разработки программного и информационного обеспечения.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Учебная дисциплина Основы программирования относится к обязательной части учебного плана и изучается на 2 курсе.

2.1. Требования к предварительной подготовке обучающегося:

Для освоения дисциплины предварител ьно необходимо изучение дисциплин

Основы информатики

Информационные технологии

2.2. Дисциплины (модули) и практики, для которых освоение данной дисциплины (модуля) необходимо как предшествующее:

Результаты освоения дисциплины «Основы программирования» являются базой для прохождения обучающимися производственной практики: технологической (проектно-технологической) и преддипломной, а так же используется для подготовки ВКР.

Развитие у обучающихся навыков командной работы, межличностной коммуникации, принятия решений, лидерских качеств обеспечивается чтением лекций, проведением занятий, содержание которых разработано на основе результатов научных исследований, проводимых Университетом, в том числе с учетом региональных особенностей профессиональной деятельности выпускников и потребностей работодателей.

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ, СООТНЕСЕННЫЕ С ПЛАНИРУЕМЫМИ РЕЗУЛЬТАТАМИ ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

В результате освоения дисциплины обучающийся по программе бакалавриата должен овладеть:

- Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения (ОПК-2)

Планируемые результаты обучения по дисциплине

Формируемая компетенция	Планируемые результаты обучения	Код результата обучения
Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения (ОПК-2)	**Знать:**	
	Базовые принципы процедурного программирования	ОПК-2-З1
	Базовые принципы объектно-ориентированного программирования	ОПК-2-З2
	Базовые принципы визуального программирования	ОПК-2-З3
	Архитектуру и возможности языков высокого уровня	ОПК-2-З4
	Базовые возможности интегрированных сред разработки программных комплексов	ОПК-2-З5
	Языки программирования и современные среды программирования	ОПК-2-З6
	Уметь:	
	Работать в изучаемой среде программирования	ОПК-2-У1
	Реализовывать стандартные алгоритмы в виде программных комплексов	ОПК-2-У2
	Разрабатывать программные комплексы с учетом основных требований информационной безопасности	ОПК-2-У3
	Разрабатывать интерфейс для консольных приложений	ОПК-2-У4
	Разрабатывать интерфейс для Windows-приложений	ОПК-2-У5
	Использовать алгоритмы обработки информации для различных приложений	ОПК-2-У6
	Владеть:	
	Владеть навыками работы в изучаемой среде программирования	ОПК-2-В1
	Владеть навыками разработки проектов в изучаемой среде программирования	ОПК-2-В2
	Владеть навыками разработки многооконных приложений	ОПК-2-В3
	Владеть навыками работы с локальными базами данных	ОПК-2-В4
	Методологией решения практических задач	ОПК-2-В5
	Одним из языков программирования	ОПК-2-В6

4. ОБЪЕМ ДИСЦИПЛИНЫ В ЗАЧЕТНЫХ ЕДИНИЦАХ С УКАЗАНИЕМ КОЛИЧЕСТВА АКАДЕМИЧЕСКИХ ЧАСОВ, ВЫДЕЛЕННЫХ НА КОНТАКТНУЮ РАБОТУ ОБУЧАЮЩИХСЯ С ПРЕПОДАВАТЕЛЕМ (ПО ВИДАМ УЧЕБНЫХ ЗАНЯТИЙ) И НА САМОСТОЯТЕЛЬНУЮ РАБОТУ ОБУЧАЮЩИХСЯ

Общая трудоемкость дисциплины составляет 3 зачетных единицы (108 часа).

№	Семестр	Общая трудоёмкость		В том числе контактная работа с		Контроль	Сам. работа	Форма промежуточной аттестации
		В з.е.	В часах	всего	Пр			

1	4	3	108	54	52		54	Зачет

Распределение учебного времени по темам и видам учебных занятий очная форма обучения

№	Наименование разделов, тем учебных занятий	Всего часов	Контактная работа с преподавател		Сам. раб.	Формируемые результаты обучения
			Всего	Пр		
Алгоритмизация и основы программирования.						
1.	Алгоритмизация и основы программирования.	6	4	4	2	ОПК-2-З1, ОПК-2-У2
Язык программирования С. Реализация линейных алгоритмов						
2.	Язык программирования С. Реализация линейных алгоритмов.	6	4	4	2	ОПК-2-З4, ОПК-2-З6, ОПК-2-У2
Реализация циклических алгоритмов.						
3.	Реализация циклических алгоритмов.	6	4	4	2	ОПК-2-З4, ОПК-2-З6, ОПК-2-У2
Статические и динамические массивы.						
4.	Статические и динамические массивы.	8	4	4	4	ОПК-2-З2, ОПК-2-З4, ОПК-2-У1, ОПК-2-У2
Пользовательские функции.						
5.	Пользовательские функции.	8	4	4	4	ОПК-2-З6, ОПК-2-В6
Принципы работы с файлами.						
6.	Принципы работы с файлами.	8	4	4	4	ОПК-2-У4, ОПК-2-У6, ОПК-2-В1, ОПК-2-В6
Принципы работы со строками						
7.	Принципы работы со строками	8	4	4	4	ОПК-2-З1, ОПК-2-У1, ОПК-2-У6, ОПК-2-В1
Организация интерфейса пользователя.						
8.	Организация интерфейса пользователя.	6	4	4	2	ОПК-2-З3, ОПК-2-У4, ОПК-2-У5, ОПК-2-В2
Основные понятия объектно-ориентированного подхода к программированию.						
9.	Основные понятия объектно-ориентированного подхода к программированию.	6	4	4	2	ОПК-2-З2, ОПК-2-З6, ОПК-2-В6
Инициализация объектов. Конструкторы и деструкторы. Перегрузка операций.						
10.	Инициализация объектов. Конструкторы и деструкторы. Перегрузка операций.	4	2	2	2	ОПК-2-З2, ОПК-2-З3, ОПК-2-У1, ОПК-2-В1, ОПК-2-В2
Принцип наследования и его реализация средствами языка программирования.						

11.	Принцип наследования и его реализация средствами языка программирования.	8	4	4	4	ОПК-2-З2, ОПК-2-З6, ОПК-2-У1, ОПК-2-В2
Шаблоны. Библиотека STL.						
12.	Шаблоны. Библиотека STL.	4	2	2	2	ОПК-2-З4, ОПК-2-В2, ОПК-2-В3
Обработка исключительных ситуаций.						
13.	Обработка исключительных ситуаций.	6	4	4	2	ОПК-2-З6, ОПК-2-У2, ОПК-2-У3, ОПК-2-В2
Разработка Windows - приложений.						
14.	Разработка Windows - приложений.	8	4	4	4	ОПК-2-З3, ОПК-2-З5, ОПК-2-У2, ОПК-2-У5, ОПК-2-В2, ОПК-2-В4, ОПК-2-В5
Обзор современных средств и технологий программирования.						
15.	Обзор современных средств и технологий программирования.	2			2	ОПК-2-З6, ОПК-2-У3, ОПК-2-В5, ОПК-2-В6
Промежуточная аттестация(Зачет)						
16.	Зачет	14	2		12	

5. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ, СТРУКТУРИРОВАННОЕ ПО ТЕМАМ (РАЗДЕЛАМ)

Тема 1. **Алгоритмизация и основы программирования. .**

Понятие алгоритма. Линейные алгоритмы, алгоритмы с ветвлением, циклические алгоритмы. Представление алгоритмов в графическом виде (блок-схема) и в псевдокоде. Элементарные алгоритмические конструкции.

Типовые алгоритмы – суммирование, поиск максимума (минимума). Алгоритмы сортировки – подсчетом, методом вставки, методом пузырька. Алгоритм быстрой сортировки. Алгоритмы поиска – последовательный поиск, ступенчатый поиск, бинарный поиск.

Динамические структуры данных: списки, очереди, стек.

Методология разработки алгоритма. Оценка эффективности алгоритма. Качество программного обеспечения. Доказательное программирование.

Тема 2. **Язык программирования С. Реализация линейных алгоритмов. .**

Алфавит языка, лексемы. Ключевые слова и идентификаторы. Типы данных. Директивы препроцессора include и define. Понятие функции. Структура программы.

Переменные и константы. Глобальные и локальные переменные. Область видимости переменных. Функции ввода/вывода. Библиотечные функции.

Операция присваивания. Арифметические выражения и операции. Преобразование типов данных. Операции отношения, условные выражения и логические операции. Операторы ветвления: условный оператор if. Оператор множественного выбора switch. Конструкции case и default. Оператор прерывания break. Передача управления: оператор безусловного перехода goto.

Тема 3. **Реализация циклических алгоритмов. .**

Циклы с предусловием и постусловием. Реализация циклов с помощью операторов ветвления и передачи управления. Операторы цикла while, do while, for. Взаимное приведение циклов for и while. Оператор продолжения continue. Прерывание циклов.

Вложенные циклы. Понятие об итерации. Рекурсивные и циклические алгоритмы.

Тема 4. **Статические и динамические массивы. .**

Массивы как однородные статические структуры данных. Числовые массивы. Алгоритмы обработки массивов: суммирование, поиск максимума (минимума), сортировка, поиск. Массивы различной размерности. Заполнение и инициализация массивов. Многомерные массивы. Алгоритмы работы с матрицами.

Указатели. Динамические массивы. Арифметика указателей. Связь между массивами и указателями.

Строки. Обработка строк как массивов символов. Библиотечные функции обработки строк. Массивы строк.

Тема 5. **Пользовательские функции. .**

Библиотечные и пользовательские функции. Прототип и описание функции. Возвращаемое значение. Передача параметров по значению и по адресу. Ссылки. Массивы в качестве параметров. Параметры со значениями по умолчанию. Функции в качестве параметров. Понятие функтора. Рекурсия. Перегрузка функций.

Тема 6. **Принципы работы с файлами. .**

Файлы. Типы файлов. Организация работы с файлами. Библиотечные функции, предназначенные для работы с файлами.

Тема 7. **Принципы работы со строками**

.

Строки C++. Обработка строк как массивов символов. Библиотечные функции обработки строк. Класс string. Примеры работы со строками.

Тема 8. **Организация интерфейса пользователя. .**

Организация и средства человеко-машинного интерфейса. Работа с экраном в текстовом режиме Задание цвета. Интерфейс командной строки. Горячие клавиши. Пассивное меню. Активное меню. Создание формы ввода на экран.

Тема 9. **Основные понятия объектно-ориентированного подхода к программированию.**

Понятие инкапсуляции, полиморфизма, наследования, модульности и абстракции объектов.

Понятие класса и объекта. Объявление класса. Данные-члены (свойства) и функции-члены (методы). Доступ к членам класса: открытые, закрытые и защищенные члены класса. Объекты. Обращение к членам объекта. Дружественные классы и функции. Передача объекта в качестве параметра функции.

Тема 10. **Инициализация объектов. Конструкторы и деструкторы. Перегрузка**

Конструкторы. Свойства конструкторов. Конструктор по умолчанию. Параметры конструкторов. Конструктор преобразования. Деструктор. Вызов деструктора. Виртуальный деструктор. Указатель this. Оператор присваивания. Конструктор копий. Поразрядное (буквальное) и развернутое копирование.

Понятие перегрузки. Перегрузка функций. Перегрузка операций. Перегружаемые операции. Операции, не допускающие перегрузки.

Перегрузка унарных и бинарных операций. Перегрузка операций как методов класса и как дружественных функций. Сравнение способов перегрузки операций.

Тема 11. **Принцип наследования и его реализация средствами языка**

программирования. .

Базовый и производный классы. Режимы доступа к членам базового класса. Конструкторы и деструкторы производных классов. Полиморфизм. Виртуальные методы.

Множественное наследование. Использование виртуальных базовых классов. Чисто виртуальная функция. Абстрактные классы.

Тема 12. **Шаблоны. Библиотека STL. .**

Перегрузка функций. Шаблоны функций. Создание шаблонов функций. Классы-шаблоны. Создание и использование шаблонов классов.

Библиотека STL. Алгоритмы и контейнеры. Вектор, список, стек, очередь.

Тема 13. **Обработка исключительных ситуаций.**

.

Понятие исключения. Генерация исключения, обработчики исключений. Абсолютный обработчик. Стандартный обработчик и его перегрузка.

Исключения в классах. Исключения в функциях.

Тема 14. **Разработка Windows - приложений. .**

Структура и выполнение Windows –программ. Инициализация окон. Обработка сообщений. Управление памятью. Классы приложений. Классы-шаблоны и виртуальные функции для работы с Windows. Окно и его ресурсы. Методы создания ресурсов окна.

Тема 15. **Обзор современных средств и технологий программирования. .**

Методика проектирования, ориентированная на потоки данных. Визуальное программирование: компоненты и события, обработка событий. Многооконный интерфейс и средства его создания. Средства автоматизации программирования (CASE - технологии).

Тема 16. **Зачет .**

1. Какие существуют способы описания алгоритмов?
2. Что такое разветвляющийся алгоритм? Приведите пример.
3. В каких случаях необходима циклическая обработка данных? Приведите пример.
4. Для чего служат директивы препроцессора?
5. В чем отличия между интерпретатором и компилятором?
6. Что такое язык программирования высокого уровня. Приведите пример.
7. Что такое язык программирования низкого уровня. Приведите пример.
8. Что содержит заголовочный файл?
9. Поясните концепцию типов данных.
10. В чем отличие констант от переменных?
11. Что такое глобальная переменная?
12. В чем разница между значением и именем переменной?
13. Что такое указатель?
14. Какие операции применяются с указателями?
15. Зачем нужны операции преобразования типа? Приведите пример.
16. Перечислите операторы выбора.
17. Почему в операторе множественного выбора (switch) необходимо использовать оператор break?
18. Зачем в операторе множественного выбора применяется ключевое слово default?
19. Перечислите операторы цикла.
20. В чем отличие между инициализацией и присваиванием?
21. Перечислите арифметические операции.
22. Перечислите операции отношения.
23. Перечислите логические операции.
24. Поясните концепцию модульного программирования.

25. Что такое массив? Приведите пример объявления массива.
26. Как можно инициализировать массив?
27. Какие методы упорядочивания массива Вы знаете?
28. Какие методы поиска в массиве Вы знаете?
29. Что такое динамический массив?
30. Что такое многомерный массив? Приведите пример.
31. Какие поразрядные (побитовые) операции Вы знаете?
32. Для чего применяются манипуляторы потоков?
33. Как можно форматировать потоки?
34. Какие классы окон вы знаете?
35. Какие методы класса приложения Вы знаете?
36. Зачем нужны процедуры (функции) обработки событий?
37. Какие виды ресурсов Вы знаете?
38. Что такое контекст устройства?
39. Какие методы контекста устройства Вы знаете?
40. Какие параметры пера можно изменить?
41. Какие параметры кисти можно задать?
42. Что такое графический примитив?
43. Что такое проект?
44. Какие стандартные типы проектов Вы знаете?
45. Перечислите основные файлы проекта в изучаемой в курсе среде программирования.
46. Что такое компонент.
47. Какие компоненты Вам известны?
48. Что такое свойства компонента?
49. Что такое событие?
50. Чем отличается визуальный компонент от невизуального?
51. Какие стандартные диалоги Вам известны?
52. Члены класса.
53. Конструкторы. Конструктор по умолчанию. Конструктор преобразования.
54. Деструктор.
55. Операция присваивания.
56. Конструктор копий.
57. Методы класса.
58. Дружественные функции.
59. Методика проектирования, ориентированная на потоки данных.
60. Визуальное программирование: компоненты и события, обработка событий.

Планы практических занятий

Тема 1. **Алгоритмизация и основы программирования.. .**
Время - 4 час.
Основные вопросы:
Раработка алгоритмов.

Тема 2. **Язык программирования С. Реализация линейных алгоритмов.. .**
Время - 4 час.
Основные вопросы:
1. Язык программирования С и С++. История возникновения. Алфавит языка.
2. Основные типы данных. Структура программы. Стандартные библиотеки.
3. Основные операторы ввода-вывода (printf, scanf).
4. Ввод – вывод данных в языке С++ (cin, cout).
5. Реализация линейных алгоритмов на С++.
6. Реализация разветвляющихся алгоритмов.

Тема 3. **Реализация циклических алгоритмов.. .**
Время - 4 час.
Основные вопросы:
1. Операторы цикла с предусловием и постусловием. Синтаксис, структура операторов
2. Оператор с предусловием for. Примеры решения задач.
3. Оператор с предусловием while. Примеры решения задач.
4. Оператор с постусловием do-while. Примеры решения задач.

Тема 4. **Статические и динамические массивы.. .**
Время - 4 час.
Основные вопросы:

1. Массивы, определение. Одномерные, двумерные массивы.
2. Основные операции: ввод, обработка, вывод элементов массива.
3. Статические массивы.
4. Указатели, основные типы и действия над указателями.
5. Динамические массивы.

Тема 5. **Пользовательские функции.. .**
Время - 4 час.
Основные вопросы:
1. Понятие функций в C++.
2. Пользовательские функции: структура, форма записи.
3. Формальные и фактические параметры, локальные и глобальные переменны.
4. Вызов функции по значению и по ссылке.
5. Примеры разработки программ.
6. Рекурсивные функции.

Тема 6. **Принципы работы с файлами.. .**
Время - 4 час.
Основные вопросы:
1. Определение файла. Физический и программный файл.
2. Виды файлов: типизированные, текстовые.
3. Классификация файлов по типу доступа: прямого, последовательного доступа.
4. Основные операции с файлами.
5. Обработка файлов

Тема 7. **Принципы работы со строками**
. .
Время - 4 час.
Основные вопросы:
1. Обработка строк как массивов символов.
2. Библиотечные функции обработки строк.
3. Примеры работы со строками.

Тема 8. **Организация интерфейса пользователя.. .**
Время - 4 час.
Основные вопросы:
1. Понятие пользовательского интерфейса (ПИ).
2. Виды ПИ.
3. Основные элементы ПИ.
4. Разработка ПИ.

Тема 9. **Основные понятия объектно-ориентированного подхода к программированию.. .**

Время - 4 час.

Основные вопросы:

1.Объявление и применение классов в программах.

Тема 10. **Инициализация объектов. Конструкторы и деструкторы. Перегрузка операций.. .**

Время - 2 час.

Основные вопросы:

1. Основные члены класса: конструкторы, деструкторы.
2. Назначение и описание конструкторов и деструкторов.
3. Свойства конструкторов и деструкторов.
4. Использование конструкторов и деструкторов при разработке программ

Тема 11. **Принцип наследования и его реализация средствами языка программирования.. .**

Время - 4 час.

Основные вопросы:

1. Понятие наследования, свойства.
2. Единичное и множественное наследование.
3. Использование механизма наследования при разработке программ

Тема 12. **Шаблоны. Библиотека STL.. .**

Время - 2 час.

Основные вопросы:

1. Перегрузка функций. Шаблоны функций.
2. Классы-шаблоны. Создание и использование шаблонов классов.
3. Библиотека STL.
4. Алгоритмы и контейнеры.
5. Вектор, список, стек, очередь.

Тема 13. **Обработка исключительных ситуаций.**

. .

Время - 4 час.

Основные вопросы:

1. Понятие исключения, исключительной ситуации.
2. Обработка исключительных ситуаций try – catch. Структура оператора.
3. Использование обработки исключительных ситуаций при разработке программ.

Тема 14. **Разработка Windows - приложений.. .**

Время - 4 час.

Основные вопросы:

1. Понятие Windows – приложения, отличительные особенности.
2. Способы разработки Windows – приложений.
3. Разработка программ с использованием Windows – приложений.
4. Отладка программ - Windows – приложений.

6. ПЕРЕЧЕНЬ УЧЕБНО-МЕТОДИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

6.1.1. Основные категории учебной дисциплины для самостоятельного изучения:

Язык программирования — формальный язык, предназначенный для записи компьютерных программ. Язык программирования определяет набор лексических, синтаксических и семантических правил, определяющих внешний вид программы и действия, которые выполнит исполнитель (обычно — ЭВМ) под её управлением.

Алфавит языка - основные символы языка— буквы, цифры и специальные символы.

Лексема — последовательность допустимых символов языка программирования, имеющая смысл для транслятора.

Идентификатор (ID) – это имя программного объекта (константы, переменной, метки, типа, функции и т.д.). В идентификаторе могут использоваться латинские буквы, цифры и знак подчеркивания; первый символ ID – не цифра; пробелы внутри ID не допускаются.

Ключевые слова - идентификаторы, смысл которого зафиксирован правилами языка программирования и который используется для распознавания предложений в программе

Псевдокод — компактный (зачастую неформальный) язык описания алгоритмов, использующий ключевые слова императивных языков программирования, но опускающий несущественные подробности и специфический синтаксис. Псевдокод обычно опускает детали, несущественные для понимания алгоритма человеком. Такими несущественными деталями могут быть описания переменных, системно-зависимый код и подпрограммы.

Блок-схема — распространенный тип схем (графических моделей), описывающих алгоритмы или процессы, в которых отдельные шаги изображаются в виде блоков различной формы, соединенных между собой линиями, указывающими направление последовательности.

Трансля́ция програ́ммы — преобразование программы, представленной на одном из языков программирования, в машинный код или, иногда, в язык ассемблера. Транслятор обычно выполняет также диагностику ошибок, формирует словари идентификаторов, выдаёт для печати текст программы и т. д.

Трансля́тор — программа или техническое средство, выполняющее трансляцию. программы.

Компоновщик (также реда́ктор свя́зей, линкер — от англ. link editor, linker) — программа, которая производит компоновку — принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль. Для связывания модулей, компоновщик использует таблицы имён, созданные компилятором в каждом из объектных модулей.

Компилировать — проводить трансляцию машинной программы с проблемно-ориентированного языка на машинно-ориентированный язык

Сбо́рка (англ. build) (имя существительное) — подготовленный для использования информационный продукт. Чаще всего сборка — исполняемый файл — двоичный файл, содержащий исполняемый код (машинные инструкции) программы или библиотеки.

Сбо́рка (англ. build) (глагол) — процесс получения информационного продукта из исходного кода. Чаще всего включает компиляцию и компоновку, выполняется инструментами автоматизации.

Листинг - исхо́дный код (также исхо́дный текст) — текст компьютерной программы на каком-либо языке программирования. В обобщённом смысле — любые входные данные для транслятора. Исходный код либо транслируется в исполняемый код при помощи компилятора, либо исполняется непосредственно по тексту при помощи интерпретатора.

Исполняемый код (Маши́нный код, платфо́рменно-ориенти́рованный код), маши́нный язы́к — система команд (набор кодов операций) конкретной вычислительной машины, которая интерпретируется непосредственно процессором или микропрограммами этой вычислительной машины. Компьютерная программа, записанная на машинном языке, состоит из машинных инструкций, каждая из которых представлена в машинном коде в виде т. н. опкода — двоичного кода отдельной операции из системы команд машины. Для удобства программирования вместо числовых опкодов, которые только и понимает процессор, обычно используют их условные буквенные мнемоники. Набор таких мнемоник, вместе с некоторыми дополнительными возможностями (например, некоторыми макрокомандами, директивами), называется языком ассемблера.

Инстру́кция или опера́тор (англ. statement) — наименьшая автономная часть языка программирования; команда или набор команд. Программа обычно представляет собой последовательность инструкций. Многие языки (например, Си) различают инструкцию и определение. Различие в том, что инструкция исполняет код, а определение создаёт идентификатор.

Точка входа (англ. Entry Point (EP) — точка входа) — адрес в оперативной памяти, с которого начинается выполнение программы. Другими словами — адрес, по которому хранится первая команда программы. Однако не надо путать её с «первыми командами» программы на языке высокого уровня. Например, программа на C++ начинает выполнение с функции main(), на самом

программистами для разработки программного обеспечения (ПО). Среда разработки включает в себя: текстовый редактор, компилятор и/или интерпретатор, средства автоматизации сборки, отладчик.

Отла́дка — этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки. Чтобы понять, где возникла ошибка, приходится:

Отладчик - это программный модуль, который позволяет выполнить основные задачи, связанные с мониторингом процесса выполнения результирующей прикладной программы. Этот процесс называется отладкой и включает в себя следующие основные возможности: последовательное пошаговое выполнение результирующей программы на основе шагов по машинным командам или по операторам входного языка; выполнение результирующей программы до достижения ею одной из заданных точек останова (адресов останова); выполнение результирующей программы до наступления некоторых заданных условий, связанных с данными и адресами, обрабатываемыми этой программой; просмотр содержимого областей памяти, занятых командами или данными результирующей программы.

Трассиро́вка — процесс пошагового выполнения программы. В режиме трассировки программист видит последовательность выполнения команд и значения переменных на данном шаге выполнения программы, что позволяет легче обнаруживать ошибки. Трассировка может быть начата и окончена в любом месте программы, выполнение программы может останавливаться на каждой команде или на точках останова, трассировка может выполняться с заходом в процедуры и без заходов, а также осуществляться в обратном порядке (шаг назад).

Визуальное программирование — способ создания программы для ЭВМ путём манипулирования графическими объектами вместо написания её текста. Визуальное программирование часто представляют как следующий этап развития текстовых языков программирования.

Визуальные средства разработки — как правило, под ними подразумевают средства проектирования интерфейсов или какую либо CASE-систему для быстрой разработки приложений.

Компонент — в программировании, множество классов и языковых конструкций, объединенных по общему признаку. В большинстве языков программирования нет языковых конструкций прямо отражающих понятие компонента. Компоненты реализуются с помощью стандартных конструкций, таких как классы

6.1.2. Задания для повторения и углубления приобретаемых знаний.

№	Код результата обучения	Задания
1	ОПК-2-З1	1.1. Дать определение понятию «Процедурное программирование», привести примеры.
2	ОПК-2-З1	1.2. Дать определение понятию «Структурное программирование», привести примеры.
3	ОПК-2-З2	2.1. Дать определение понятию «Объектно-ориентированное программирование», привести примеры.
4	ОПК-2-З2	2.1. Дать определение понятию «Объектно-ориентированное программирование», привести примеры.
5	ОПК-2-З3	3.1. Дать определение понятию «Визуальное программирование», привести примеры.
6	ОПК-2-З3	3.2. Привести примеры использования принципов визуального программирования в изучаемой интегрированной среде.
7	ОПК-2-З4	4.1. Объяснить основные этапы разработки алгоритмов решения научных и практических задач.
8	ОПК-2-З4	4.2. Привести примеры алгоритмов используемых при поиске данных.
9	ОПК-2-З5	5.1. Перечислить типы проектов в изучаемой интегрированной среде программирования.
10	ОПК-2-З5	5.2. Пояснить содержание основных файлов проекта, создаваемых средой программирования.
11	ОПК-2-З6	6.1. Перечислить основные синтаксические конструкции изучаемого языка программирования.

12	ОПК-2-З6	6.2. Перечислить этапы разработки проектов в изучаемой интегрированной среде программирования.

6.2. Задания, направленные на формирование профессиональных умений.

№	Код результата обучения	Задания
13	ОПК-2-У1	7.1. Создать консольное приложение, предназначенное для просмотра содержимого файла.
14	ОПК-2-У1	7.2. Создать приложение Windows, предназначенное для просмотра содержимого файла.
15	ОПК-2-У2	8.1. Написать программу, реализующую операции упорядочивания и поиска числовых данных.
16	ОПК-2-У2	8.2. Написать программу, реализующую операции упорядочивания и поиска символьной информации.
17	ОПК-2-У3	9.1. Написать программу, реализующую кодирование данных.
18	ОПК-2-У3	9.2. Написать программу, реализующую кодирование символьной информации при работе с файлами.
19	ОПК-2-У4	10.1. Создать консольное приложение, реализующее интерфейс на основе горячих клавиш.
20	ОПК-2-У4	10.2. Создать консольное приложение, реализующее интерфейс на основе меню.
21	ОПК-2-У5	11.1. Создать Windows-приложение, реализующее интерфейс на основе меню.
22	ОПК-2-У5	11.2. Создать Windows-приложение, реализующее интерфейс на основе панели инструментов.
23	ОПК-2-У6	12.1. Сформулировать требования к программному комплексу, предназначенному для обработки символьной информации..
24	ОПК-2-У6	12.2. Сформулировать требования к программному комплексу, предназначенному для обработки графической информации.

6.3. Задания, направленные на формирование профессиональных навыков, владений.

№	Код результата обучения	Задания
25	ОПК-2-В1	13.1. Создать консольное приложение.
26	ОПК-2-В1	13.2. Создать приложение Windows.
27	ОПК-2-В2	14.1. Объяснить способы организации интерфейса в консольных приложениях, привести примеры.
28	ОПК-2-В2	14.2. Объяснить способы организации интерфейса в приложениях Windows.
29	ОПК-2-В3	15.1. Объяснить способы создания многооконных приложений, привести пример.
30	ОПК-2-В3	15.2. Объяснить способы создания приложений с использованием вкладок, привести пример.
31	ОПК-2-В4	16.1. Разработать программу, реализующую обработку табличных данных отвечающих заданным объектам предметной области.
32	ОПК-2-В4	16.2. Разработать программу, реализующую обработку данных отвечающих заданным объектам предметной области с использованием базы данных.
33	ОПК-2-В5	17.1. Объяснить основные этапы разработки алгоритмов решения научных и практических задач.
34	ОПК-2-В5	17.2. Перечислить основные этапы разработки программных комплексов реализующих алгоритмы решения научных и практических задач. Привести пример.
35	ОПК-2-В6	18.1. Перечислить основные синтаксические конструкции изучаемого языка программирования.

36	ОПК-2-В6	18.2. Перечислить базовые и пользовательские типы данных изучаемого языка программирования.

7. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

7.1. Средства оценивания в ходе текущего контроля:

- письменные краткие опросы в ходе аудиторных занятий на знание категорий учебной дисциплины
- задания и упражнения, рекомендованные для самостоятельной работы;
- практические работы.
- ответы на вопросы при подготовке к зачету.

7.2. ФОС для текущего контроля:

№	Код результата обучения	ФОС текущего контроля
1	ОПК-2-З1	Задания для самостоятельной работы 1.1
2	ОПК-2-З1	Задания для самостоятельной работы 1.2
3	ОПК-2-З2	Задания для самостоятельной работы 2.1
4	ОПК-2-З2	Задания для самостоятельной работы 2.2
5	ОПК-2-З3	Задания для самостоятельной работы 3.1
6	ОПК-2-З3	Задания для самостоятельной работы 3.2
7	ОПК-2-З4	Задания для самостоятельной работы 4.1
8	ОПК-2-З4	Задания для самостоятельной работы 4.2
9	ОПК-2-З5	Задания для самостоятельной работы 5.1
10	ОПК-2-З5	Задания для самостоятельной работы 5.2
11	ОПК-2-З6	Задания для самостоятельной работы 6.1
12	ОПК-2-З6	Задания для самостоятельной работы 6.2
13	ОПК-2-У1	Задания для самостоятельной работы 7.1-7.2
14	ОПК-2-У1	Практические работы по темам 1,2
15	ОПК-2-У2	Задания для самостоятельной работы 8.1-8.2
16	ОПК-2-У2	Практические работы по темам 3,4
17	ОПК-2-У3	Задания для самостоятельной работы 9.1-9.2
18	ОПК-2-У3	Практические работы по теме 5
19	ОПК-2-У4	Задания для самостоятельной работы 10.1-10.2
20	ОПК-2-У4	Практические работы по темам 6,7
21	ОПК-2-У5	Задания для самостоятельной работы 11.1-11.2
22	ОПК-2-У5	Практические работы по темам 8
23	ОПК-2-У6	Задания для самостоятельной работы 12.1-12.2
24	ОПК-2-У6	Практические работы по темам 9
25	ОПК-2-В1	Задания для самостоятельной работы 13.1-13.2
26	ОПК-2-В1	Практические работы по темам 10
27	ОПК-2-В2	Задания для самостоятельной работы 14.1-14.2
28	ОПК-2-В2	Практические работы по темам 11
29	ОПК-2-В3	Задания для самостоятельной работы 15.1-15.2
30	ОПК-2-В3	Практические работы по темам 12
31	ОПК-2-В4	Задания для самостоятельной работы 16.1-16.2
32	ОПК-2-В4	Практические работы по темам 13
33	ОПК-2-В5	Задания для самостоятельной работы 17.1-17.2
34	ОПК-2-В5	Практические работы по темам 14
35	ОПК-2-В6	Задания для самостоятельной работы 18.1-18.2
36	ОПК-2-В6	Практические работы по темам 15

7.3 ФОС для промежуточной аттестации:

Задания для оценки знаний.

№	Код результата обучения	Задания
1	ОПК-2-З1	1. Какие существуют способы описания алгоритмов? 2. Что такое разветвляющийся алгоритм? Приведите пример. 3. В каких случаях необходима циклическая обработка данных? Приведите пример. 4. Поясните, что такое трансляция программы. 5. Для чего служат директивы препроцессора? 6. В чем отличия между интерпретатором и компилятором? 7. Поясните, что такое исполняемая программа. Приведите пример.
2	ОПК-2-З1	8. Что такое язык программирования высокого уровня. Приведите пример. 9. Что такое язык программирования низкого уровня. Приведите пример. 10. Что содержит заголовочный файл? 11. Поясните концепцию типов данных. В чем отличие констант от переменных? 12. Что такое глобальная переменная? 13. В чем разница между значением и именем переменной? 14. Что такое указатель? Какие операции применяются с указателями? 15. Зачем нужны операции преобразования типа? Приведите пример.
3	ОПК-2-З2	16. Перечислите операторы выбора. Зачем в операторе множественного выбора применяется ключевое слово default? 17. Почему в операторе множественного выбора (switch) необходимо использовать оператор break? 18. Перечислите операторы цикла. 19. В чем отличие между инициализацией и присваиванием?
4	ОПК-2-З2	20. Перечислите арифметические, логические операции, операции отношения. 21. Поясните концепцию модульного программирования. 22. Что такое массив? Приведите пример объявления, инициализации массива. 23. Какие методы упорядочивания массива Вы знаете?
5	ОПК-2-З3	24. Какие методы поиска в массиве Вы знаете? 25. Что такое динамический массив? 26. Что такое многомерный массив? Приведите пример. 27. Какие поразрядные (побитовые) операции Вы знаете? 28. Для чего применяются манипуляторы потоков? Как можно форматировать потоки?
6	ОПК-2-З3	29. Какие классы окон вы знаете? 30. Какие методы класса приложения Вы знаете? 31. Зачем нужны процедуры (функции) обработки событий? 32. Какие виды ресурсов Вы знаете? 33. Что такое контекст устройства? Какие методы контекста устройства Вы знаете? 34. Что такое графический примитив? 35. Что такое проект? Какие стандартные типы проектов Вы знаете?
7	ОПК-2-З4	36. Перечислите основные файлы проекта в изучаемой в курсе среде программирования. 37. Что такое компонент. Какие компоненты Вам известны? Что такое свойства компонента? 38. Что такое класс? Поясните понятие методы класса.

8	ОПК-2-З4	39. Что такое событие? 40. Чем отличается визуальный компонент от невизуального? 41. Какие стандартные диалоги Вам известны? 42. Конструкторы. Конструктор по умолчанию. Конструктор преобразования. 43. Деструктор. 44. Конструктор копий. 45. Дружественные функции.
9	ОПК-2-З5	46. Что такое активное меню? 47. Что такое пассивное меню? 48. Что такое пользовательский тип данных. Приведите пример. 49. Что такое структура в С? Приведите пример.
10	ОПК-2-З5	50. Что означает перечисление? 51. Что такое файл прямого доступа? 52. Какие функции работы с файлами Вы знаете? 53. В чем состоит концепция объектно-ориентированного программирования?
11	ОПК-2-З6	54. Конструкторы. Конструктор по умолчанию. Конструктор преобразования. Деструктор. 55. Операция присваивания. 56. Конструктор копий. 57. Методы класса. 58. Дружественные.
12	ОПК-2-З6	59. Методика проектирования, ориентированная на потоки данных. 60. Визуальное программирование: компоненты и события, обработка событий.

Задания для оценки умений.

№	Код результата обучения	Задания
1	ОПК-2-У1	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 7.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
2	ОПК-2-У1	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 7.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
3	ОПК-2-У2	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 8.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
4	ОПК-2-У2	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 8.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
5	ОПК-2-У3	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 9.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
6	ОПК-2-У3	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 9.2 рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.

7	ОПК-2-У4	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 10.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
8	ОПК-2-У4	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 10.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
9	ОПК-2-У5	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 11.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
10	ОПК-2-У5	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 11.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
11	ОПК-2-У6	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 12.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.
12	ОПК-2-У6	В качестве фондов оценочных средств для оценки умений обучающегося используются задания 12.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.2.), а также выполнение практических работ.

Задания, направленные на формирование профессиональных навыков, владений.

№	Код результата обучения	Задания
1	ОПК-2-В1	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 13.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
2	ОПК-2-В1	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 13.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
3	ОПК-2-В2	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 14.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
4	ОПК-2-В2	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 14.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
5	ОПК-2-В3	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 15.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.

6	ОПК-2-В3	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 15.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
7	ОПК-2-В4	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 16.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
8	ОПК-2-В4	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 16.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
9	ОПК-2-В5	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 17.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
10	ОПК-2-В5	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 17.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
11	ОПК-2-В6	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 18.1, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.
12	ОПК-2-В6	В качестве фондов оценочных средств для оценки навыков, владений, опыта деятельности обучающегося используются задания 18.2, рекомендованные для выполнения в часы самостоятельной работы (раздел 6.3.), а также практические работы, чтение лекций, дополнительной литературы, выполнение курсовой работы.

8. ПЕРЕЧЕНЬ ОСНОВНОЙ И ДОПОЛНИТЕЛЬНОЙ УЧЕБНОЙ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

а) основная литература:

1. Терехов, А. Н. Технология программирования : учебное пособие / А. Н. Терехов. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 152 c. — ISBN 978-5-4487-0070-5. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/67370.html

2. Фарафонов, А. С. Программирование на языке высокого уровня : методические указания к проведению лабораторных работ по курсу «Программирование» / А. С. Фарафонов. — Липецк : Липецкий государственный технический университет, ЭБС АСВ, 2013. — 32 c. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/22912.html

3. Программирование на языке высокого уровня C/C++ : конспект лекций / составители С. П. Зоткин. — М. : Московский государственный строительный университет, Ай Пи Эр Медиа, ЭБС АСВ, 2016. — 140 c. — ISBN 978-5-7264-1285-6. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/48037.html

б) дополнительная литература:

1. Маслянкин, В. И. Визуальное программирование : методический сборник / В. И. Маслянкин. — М. : Российский новый университет, 2010. — 40 c. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/21265.html

2. Ковалевская, Е. В. Методы программирования : учебное пособие / Е. В. Ковалевская, Н. В. Комлева. — М. : Евразийский открытый институт, 2011. — 320 c. — ISBN 978-5-374-00356-7. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/10784.html

3. Визуальное программирование на основе библиотеки MFC : методические указания к лабораторным работам по курсу «Визуальное программирование» для студентов направления 09.03.02 Информационные системы и технологии / составители А. Я. Лахов, Р. Е. Борщиков. — Саратов : Вузовское образование, 2016. — 57 c. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/28324.html

9. ПЕРЕЧЕНЬ КОМПЛЕКТОВ ЛИЦЕНЗИОННОГО И СВОБОДНО РАСПРОСТРАНЯЕМОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ИСПОЛЬЗУЕМОГО ПРИ ИЗУЧЕНИИ УЧЕБНОЙ ДИСЦИПЛИНЫ

При изучении учебной дисциплины (в том числе в интерактивной форме) предполагается применение современных информационных технологий. Комплект программного обеспечения для их использования включает в себя:

пакеты офисного программного обеспечения Microsoft Office (Word, Excel, PowerPoint), OpenOffice;

веб-браузер (Google Chrome, Mozilla Firefox, Internet Explorer др.);

электронную библиотечную систему IPRBooks;

систему размещения в сети «Интернет» и проверки на наличие заимствований курсовых, научных и выпускных квалификационных работ «ВКР-ВУЗ.РФ».

Для доступа к учебному плану и результатам освоения дисциплины, формирования Портфолио обучающегося используется Личный кабинет студента (он-лайн доступ через сеть Интернет http://lk.rosnou.ru). Для обеспечения доступа обучающихся во внеучебное время к электронным образовательным ресурсам учебной дисциплины, а также для студентов, обучающихся с применением дистанционных образовательных технологий, используется портал электронного обучения на базе СДО Moodle (он-лайн доступ через сеть Интернет https://e-edu.rosnou.ru).

10. ПЕРЕЧЕНЬ РЕСУРСОВ ИНФОРМАЦИОННО-ТЕЛЕКОММУНИКАЦИОННОЙ СЕТИ «ИНТЕРНЕТ», НЕОБХОДИМЫХ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

https://cyberleninka.ru/ научная электронная библиотека «КИБЕРЛЕНИНКА»
https://elibrary.ru/ научная электронная библиотека
http://www.gpntb.ru/ Государственная публичная научно-техническая библиотека России
http://www.iprbookshop.ru/ Электронная библиотечная система университета

11. ОБУЧЕНИЕ ИНВАЛИДОВ И ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

Изучение учебной дисциплины обучающимися инвалидами и лицами с ограниченными возможностями здоровья осуществляется в соответствии с Приказом Министерства образования и науки РФ от 9 ноября 2015 г. № 1309 «Об утверждении Порядка обеспечения условий доступности для инвалидов объектов и предоставляемых услуг в сфере образования, а также оказания им при этом необходимой помощи» (с изменениями и дополнениями), Методическими рекомендациями по организации образовательного процесса для обучения инвалидов и лиц с ограниченными возможностями здоровья в образовательных организациях высшего образования, в том числе оснащенности образовательного процесса, утвержденными Министерством образования и науки РФ 08.04.2014г. № АК-44/05вн, Положением об организации обучения студентов – инвалидов и лиц с ограниченными возможностями здоровья, утвержденным приказом ректора Университета от 6 ноября 2015 года №60/о, Положением о Центре инклюзивного образования и психологической помощи АНО ВО «Российский новый университет», утвержденного приказом ректора от 20 мая 2016 года № 187/о.

Лица с ограниченными возможностями здоровья и инвалиды обеспечиваются электронными образовательными ресурсами, адаптированными к состоянию их здоровья.

Предоставление специальных технических средств обучения коллективного и индивидуального пользования, подбор и разработка учебных материалов для обучающихся с ограниченными возможностями здоровья производится преподавателями с учетом индивидуальных психофизиологических особенностей обучающихся и специфики приема-передачи учебной информации на основании просьбы, выраженной в письменной форме.

С обучающимися по индивидуальному плану или индивидуальному графику проводятся индивидуальные занятия и консультации.

12. ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

Практические занятия проводятся в аудиториях, оборудованных экраном, проектором, компьютерами со специализированным программным обеспечением(IDE Embarcadero XE5 и Visual Studio 2010 и 2017).

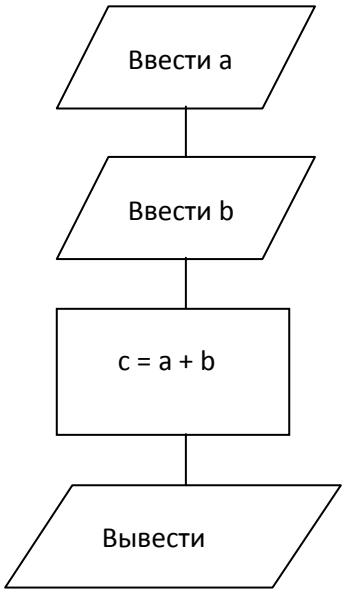
Занятия с инвалидами по зрению, слуху, с нарушениями опорно-двигательного аппарата проводятся в специально оборудованных аудиториях по их просьбе, выраженной в письменной форме.

Автор (составитель) кандидат технических наук, доцент Раскатова МВ

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ.

1. Практические упражнения могут включать в себя предложение нарисовать блок-схему алгоритма по заданному псевдокоду, и наоборот.
   Например: ввести два числа и вывести сумму:

Блок - схема



Псевдокод

1. Ввести первое число (a)
2. Ввести второе число (b)
3. Посчитать сумму (c = a + b)
4. Вывести результат

2. Структура программы на языке С выглядит следующим образом:

```
#include <conio.h>
#include <stdio.h>
    //добавить объявление и инициализацию переменных a и b .
void fun(){
    //добавить объявление и инициализацию переменных a и b .
    //с помощью функции printf вывести значения локальных переменных a и b .
    //с помощью функции printf вывести значения глобальных переменных a и b .
    //вывести значения адресов локальных и глобальных переменных a и b .
}
void main(){
    //добавить объявление и инициализацию переменных a и b .
    //вызвать функцию fun
```

// с помощью функции ***printf*** вывести значения локальных переменных ***a*** и ***b*** .

// с помощью функции ***printf*** вывести значения глобальных переменных ***a*** и ***b*** .

//вывести значения адресов локальных и глобальных переменных ***a*** и ***b*** .

getch();

}

Добавить в заготовку программы необходимые инструкции. Все переменные должны быть инициализированы различными значениями. Отладить программу и запустить её на исполнение. Сравнить значения и *порядок следования* адресов локальных и глобальных переменных.

Ответить на вопросы:

- зачем нужна директива препроцессора *#include <stdio.h>;*
- почему адреса локальных и глобальных переменных сильно различаются;
- почему адреса локальных переменных расположены в обратном порядке;
- зачем при работе в среде Borland C++ 5 нужна инструкция *getch()*;

3. Арифметические выражения и операции. Параметры функции.
1) Написать три варианта функции, вычисляющей сумму двух чисел. Результат сложения передать через параметр – указатель, через параметр - ссылку и через возвращаемое значение. Прототипы функций имеют вид:

```
void pSum(int a, int b, int* s);

void vSum(int a, int b, int& s);

int iSum(int a, int b);
```

В головной функции модуля (***main***) задать значение переменных и подсчитать соответствующие суммы, используя все варианты функции ***Sum*****.** Вывести результаты.

2) Написать программу, вычисляющую выражение:

$$\frac{a * x + \sin(b)}{4 * a + b * e^{-a}}$$

Все переменные должны быть целого типа. Обеспечить правильный результат (без потери точности). Например, при a = 1, b = 0, x = 1, программа должна выводить результат 0,25. Проверить работоспособность программы. Оформить вычисление выражения в виде функции.

4. Написать пользовательские функции, предназначенные для ввода переменной, передаваемой через параметр, вычисления факториала (n! = 1*2*3*…*n, указание - использовать рекурсию), вычисления ряда (приближение функции ***sin(x)***):

$$f(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!}$$

и функцию, осуществляющую вывод значений переменной ***x***, ряда ***f(x)*** и ***sin(x)***. В функции ***main*** помимо декларации переменных должны содержаться *только* вызовы перечисленных выше функций. *Глобальных переменных не использовать*!

Варианты: приближенное вычисление ***cos(x), e^x, ln(1+x)*** и т.п.

5. Алгоритм с ветвлением. Функции – возвращаемое значение.

1) Написать программу, проверяющую значение числового пароля, вводимого пользователем. При правильном значении пароля программа должна выводить «hello!», в противном случае «go out!». Оформить проверку пароля в виде отдельной функции.

2) Написать программу, запрашивающую значения трех чисел – длин отрезков и проверяющую, возможно ли из этих отрезков составить треугольник. Оформить проверку в виде отдельной функции.

Указание: в треугольнике сумма длин любых двух сторон больше третьей.

3) Оформить в виде отдельной функции вычисление выражения:

$$f(x)=\begin{cases}\dfrac{a*x+b}{a-b}, & если \quad a\neq b\\ 0, & если \quad a=b \quad и \quad a=0\\ \dfrac{b*x}{a}, & если \quad a=b \quad и \quad a\neq 0\end{cases}$$

Написать программу, запрашивающую значения трех чисел и выводящую ***f(x).***

6. *В данном задании для организации циклов использовать операторы* ***if*** и ***goto.***

1) Написать программу, выводящую на экран пассивное меню:

Enter value from one to four

brown - 1

blond - 2

red - 3

black – 4

В случае если введенное пользователем значение не соответствует ни одному пункту меню,

приложение должно предложить повторно выбрать допустимый пункт.

2) Добавить реакцию на выбранный пункт:

- если выбрана 1, вывести «have an alibi»;
- 2 – «is dead»;
- 3 – «witness»;
- 4 – «wanted»;
- в остальных случаях – «bald»

3) Добавить в программу запрос на повторный ввод данных: ***once more? (y/n)***

и в случае, если пользователь нажмет клавишу ***"y"***, возврат к пункту 1) задания.

4) Оформить вывод меню и обработку выбранного значения отдельными функциями.

7. *В данном задании для организации циклов использовать операторы* ***while*** и ***do while.***

1) Написать программу проверки пароля. В случае если пароль не верен, приложение должно повторно запрашивать пароль.

2) Переписать программы предыдущего (13–го) задания с помощью инструкций ***while*** и ***do while.***

3) Написать программу суммирования и подсчета арифметического среднего вводимых с клавиатуры ***n*** чисел. Число ***n*** вводится с клавиатуры пользователем.

4) Написать программу выбора максимального или минимального значения из вводимых с клавиатуры ***n*** чисел. Число ***n*** вводится с клавиатуры пользователем.

5) Написать варианты программ суммирования и поиска максимального или минимального значения из вводимых с клавиатуры чисел для случая, когда их количество неизвестно (***n*** не задается), а подсчет заканчивается, если очередное введенное число оказалось равно 2006. Число 2006 не должно учитываться при вычислении суммы и среднего и поиске максимума или минимума.

8. *В данном задании для организации циклов использовать оператор* ***for.***

1)Переписать программы предыдущего задания (задание 14, пункты с 3 по 5) с помощью оператора ***for***.

2*)Используя только целые типы данных и арифметические операции написать программу, запрашивающую от пользователя целое число и выводящую целое число, записанное в обратном порядке цифр. Например, ввели 134, получили 431.

3*)Используя только целые типы данных и арифметические операции написать программу, находящую наибольший общий делитель двух целых чисел (применить алгоритм Евклида).

4*)Используя только целые типы данных и арифметические операции написать программу перевода десятичного целого в двоичное представление.

9. Функция ***sin(x)*** может быть представлена в виде ряда:

$$f(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \text{K} + \frac{x^n}{n!} + \text{K}$$

1)Написать пользовательскую функцию, вычисляющую значение этого ряда в точке ***x*** с заданной точностью **ε** (то есть последний член ряда $\left|\frac{x^n}{n!}\right| \le \varepsilon$).

2) Написать пользовательскую функцию, вычисляющую сумму первых n членов этого ряда в точке ***x***.

3)Написать функцию, выводящую результаты вычислений в виде таблицы в диапазоне от x_0 до $x_к$ с шагом Δx:

x	f(x)	sin(x)
0.00	0.0000000	0.0000000
0.10	0.0998334	0.0998334
…..	…….	…….

В последнем столбце должно приводиться значение, рассчитанное с помощью стандартной библиотечной функции ***sin(x)***.

4)Написать и отладить программу, запрашивающую пользователя диапазон, шаг и желаемую точность вычислений (или количество членов ряда) и выдающую результат в виде таблицы. Ввод всех параметров оформить в виде отдельной функции.

Глобальных переменных не использовать!

Варианты: приближенное вычисление ***cos(x), e^x, ln(1+x)*** и т.п

10. При нажатии клавиш клавиатуры функция ***getch()*** возвращает целое число код клавиши. Для алфавитно-цифровых клавиш это число не нулевое, однако для других клавиш, например функциональных или клавиш управления курсора результатом будет ноль. Для определения (расширенного) кода таких клавиш необходим повторный вызов функции ***getch()***. Конструкция

```
int c;

…

while((c=getch())==0);
```

позволит получить не нулевой код независимо от типа клавиши.

1)Написать программу, выводящую в цикле символьное значение нажатой клавиши и ее код. Окончание программы – нажатие клавиши ***Esc*** (код – 27). Найти и законспектировать коды клавиш управления курсора, пробела, табуляции и клавиши ввода (***Enter***).

2)Используя операторы цикла и ***switch***, написать программу, обрабатывающую выбор пользователем клавиш клавиатуры и сообщающую, какая клавиша была нажата:

«Вы нажали …»

для клавиш, коды которых найдены в предыдущем пункте задания. Для остальных клавиш должно выдаваться сообщение «Смотри что жмешь!!!». Выход из программы – ***Esc.*** Все сообщения должны быть *на русском языке*.

Указание: для перекодировки кириллицы из ANSI (Windows) в ASCII (DOS) можно использовать функцию

```
void   CharToOem(char* источник_ANSI, char* результат_ASCII),
```

определенную в заголовочном файле ***windows.h*** или утилиту Fconvert в Borland C++.

3)Написать программу, выводящую в левый верхний угол окна *активное* вертикальное меню. Выбор пунктов меню должен осуществляться курсорными клавишами ↑ и ↓. Нажатие остальных клавиш должно приводить к выдаче в правом углу сообщений, аналогично предыдущему пункту задания. Выбранный пункт меню выделить инверсными цветами. Выход из программы – ***Esc.*** Все сообщения и пункты меню должны быть *на русском языке*.

Указания: Использовать функции ***clrscr, clreol, textcolor, textbackground, gotoxy, cputs*** и т.п., определенные в ***conio.h***. Алгоритм может иметь следующий вид:

1. перекодировка кириллицы
2. начало внешнего цикла
3. задание цветов меню
4. начало внутреннего цикла – вывод меню

5. выбор строки окна
6. вывод очередного пункта меню
7. если меню нарисовано не полностью – переход к п.4
8. выбор инверсных цветов
9. выбор строки окна
10. вывод выбранного пункта меню инверсными цветами
11. считывание кода нажатой клавиши
12. действия, в зависимости от кода клавиши – ***switch***
13. если не нажата клавиша ***Esc*** – возврат к п.2
14. выход

4*)Написать программу, выводящую в первой строке окна *активное* горизонтальное меню. Выбор пунктов меню должен осуществляться курсорными клавишами ← и →.

11. Обработка одномерных массивов чисел.

1)Написать функции ввода/вывода числовых массивов, суммирования их элементов и поиска максимума или минимума. В функции через параметры должны передаваться массив и количество элементов, подлежащих обработке. Написать программу, использующую эти функции.

2)Написать программу, осуществляющую упорядочивание элементов массива.

3)Написать программу поиска в массиве элемента с заданным значением. Программа должна выводить индекс найденного элемента массива или **–1.**

4)Оформить решение задач сортировки и поиска в массиве в виде функций.

5*)Написать программу, имитирующую работу с иерархическим многоуровневым меню. Выбор пунктов меню должен осуществляться курсорными клавишами, клавиша ***Esc*** – должна возвращать пользователя к предыдущему уровню меню, клавиша ввода (***Enter***) – переход к вложенному меню или вывод сообщения**.**

12. Обработка двумерных массивов чисел.

1)Написать функции ввода/вывода двумерных числовых массивов.

2)Написать программу умножения матриц.

3)Написать программу транспонирования матрицы, без использования дополнительных массивов (т.е. саму в себя).

4*)Написать программу расчета определителей произвольного порядка.

Указание: использовать разложение определителя по строке.

13. Написать *пользовательские* функции работы со строками. Прототипы функций:

```
int strlen(char* s);                    // длина строки

char* strcpy(char* out,char* in);       // копирование

char* strcat(char* out,char* in);       // конкатенация

char* strchr(char* s,char c);           // поиск символа в строке

char* strstr(char* s, char* str);       // поиск вхождения
```

В головной функции продемонстрировать работоспособность Ваших функций – задать строку, скопировать её, дописать текст в конец строки и вывести результаты.

14. Стандартные функции обработки строк. Массивы строк.

1)Заменить в программе, созданной на предыдущем задании пользовательские функции на библиотечные (***string.h***). Проверить работу программы.

2)Написать функцию проверки пароля, состоящего из произвольных символов.

3)Написать программу подсчета количества заданного пользователем символа или вхождения в строке (***strchr*** или ***strstr*****).**

4)Написать программу подсчета количества слов в строке (***strtok*****).**

5)Написать функции упорядочивания массива строк и поиска строки в массиве. Функция поиска должна выводить индекс найденной в массиве строки или **–1.**

6*)Пользователь вводит строку, содержащую арифметическое выражение. Например, 12+3*4-(2+8)/2. Проверить корректность выражения. Посчитать результат.

15. Структуры, массивы структур.

1) Описать структуру «Студент (номер, имя, средний балл)».

2) Написать функции ввода/вывода массива структур «Студент».

3)Написать функцию упорядочивания массива структур «Студент» по среднему баллу.

4) Написать функцию поиска в массиве структур «Студент» нужной записи. Функция должна возвращать индекс найденной записи или **–1.**

5) Написать и отладить программу, использующую эти функции.

6*)Разработать структуру – фрейм меню. Написать программу, имитирующую работу с иерархическим многоуровневым меню. Выбор пунктов меню должен осуществляться курсорными клавишами, клавиша ***Esc*** – должна возвращать пользователя к предыдущему уровню меню, клавиша ввода (***Enter***) – переход к вложенному меню или вывод сообщения**.**

16. Используя функции, написанные на предыдущем занятии и алгоритмы работы с активным меню (рассмотренные в задании 16), разработать базу данных «Студент». Программная реализация должна предусматривать решение задач ввода и удаления данных, просмотр, сортировку и поиск. Активное меню должно иметь вид:

"Открыть "

"Сохранить "

"Добавить "

"Удалить "

"Просмотр "

"Сортировка"

"Найти "

"Выйти "

Функции открытия и сохранения базы данных будут рассмотрены позднее. Выход из программы продублировать клавишей ***Esc.*** Нажатие непредусмотренных клавиш должно приводить к звуковому сигналу - ***puts("\a")*****.**

Указание: Просмотр записей и удаление выбранной записи можно осуществлять клавишами **←, →** и **Del**, которые должны становиться доступными в режимах поиска и удаления. Соответствующие этим режимам функции желательно подробно разобрать с преподавателем.

17. Доработка базы данных «Студент».

    1)Написать функции записи в текстовый файл и чтения из файла массива структур «Студент».

    2)Написать функции записи в бинарный файл и чтения из файла массива структур «Студент».

    3)Написать функцию чтения из файла произвольного доступа записи «Студент» с переданным через параметр номером.

    4) Доработать базу данных «Студент», дополнив её функциями открытия и сохранения.

18. Рекурсия, параметры по умолчанию, перегрузка, шаблон функции. Во всех примерах задания необходимо написать программу, иллюстрирующую работу функций.

    1)Не используя циклы написать функцию, вычисляющую сумму натуральных чисел **1+2+…+n**. Число **n** является параметром функции.

Указание: применить рекурсию.

2)Написать функцию, вычисляющую сумму **n** элементов массива начиная с элемента **n0.** Переменные **n** и **n0** – параметры функции, причем параметр **n0** должен иметь значение по умолчанию, равное **0.**

3)Написать несколько перегруженных функций, выдающих сообщение о типе параметра функции. Например:

```
void message(char* str){
printf(“Это строка - %s”,str);
}
```

4)Написать шаблон функции, возвращающей сумму значений двух параметров функции. В случае если параметрами являются строки, результатом считать конкатенацию.

19. Описать структуру Студент, с полями: имя, оценка, номер. Объявить массив структур – Группа. Прочитать данные из созданного в блокноте текстового файла. Упорядочить данные. Записать в бинарный файл.
20. Написать программу, считывающую информацию о студенте из созданного на предыдущем занятии бинарного файла по номеру записи. Ввести имя студента, найти нужную запись в файле и вывести всю информацию (имя, оценка, номер) об искомом студенте.
21. Описать класс Студент, с полями: имя, оценка, номер. Поле оценка должно быть скрытым. В классе определить несколько конструкторов, методы доступа к скрытому полю, с проверкой корректности оценки, и методы ввода/вывода на консоль. Объявить массив структур – Группа. Ввести данные с клавиатуры и записать в файл.
22. В определение класса Студент добавить конструктор копий, операцию присваивания и деструктор. Объявить массивы структур – «Первый курс» и «Второй курс». Прочитать данные из файла, созданного на предыдущем занятии в таблицу «Первый курс», скопировать во вторую таблицу и вывести данные из таблицы «Второй курс».

23. В классе Студент перегрузить операции «меньше», «равно», помещения в потоки и извлечение из потока (<< и >>). Объявить массив структур – Группа. Заполнить таблицу Группа. Упорядочить данные. Ввести имя студента, найти нужную запись в файле и вывести всю информацию (имя, оценка, номер) об искомом студенте.
24. Описать класс комплексных чисел. В классе определить арифметические операции.
25. Описать структуру Студент, с полями: имя, оценка, номер. Поле имя должно быть указателем. В классе определить несколько конструкторов, методы ввода/вывода на консоль, конструктор копий, операция присваивания и деструктор. Объявить массивы структур – «Первый курс» и «Второй курс». Заполнить таблицу «Первый курс», скопировать во вторую таблицу и вывести данные из таблицы «Второй курс».
26. Описать структуру Студент, с полями: имя, оценка, номер. Поле имя должно быть объектом стандартного класса string. В классе определить несколько конструкторов, методы ввода/вывода на консоль и операции «меньше», «равно». Объявить массив структур – Группа. Заполнить таблицу Группа.
27. Описать класс Аспирант, являющийся наследником класса Студент. В наследнике добавит поле – шеф и перегрузить операции ввода вывода родительского класса.
28. Написать шаблоны функций поиска в массиве максимума, искомого значения и упорядочивания массива.
29. Написать программу, порождающую исключения различных типов при вычислениях частного двух вводимых с клавиатуры чисел, корня из числа и попытке открытия на чтение не существующего файла. Описать обработчики исключений соответствующих типов. Добавить абсолютный обработчик.
30. Создать приложение Windows типа “Hello World”
31. Создать приложение Windows – калькулятор.
32. Создать приложение Windows – текстовый редактор.