

МИНОБРНАУКИ РОССИИ
Ярославский государственный университет им. П.Г. Демидова

Кафедра компьютерной безопасности и математических методов обработки информации

УТВЕРЖДАЮ

Декан математического факультета



Нестеров П.Н.

21 мая 2024 г.

Рабочая программа дисциплины
Практикум по основам программирования

Направление подготовки (специальности)
01.03.02 Прикладная математика и информатика

Направленность (профиль)
«Прикладное программирование и информационные технологии»

Форма обучения очная

Программа рассмотрена
на заседании кафедры
от 26 апреля 2024 г., протокол № 8

Программа одобрена НМК
математического факультета
протокол № 9 от 3 мая 2024 г.

1. Цели освоения дисциплины

Дисциплина "Практикум по основам программирования" нацелена на подготовку студентов к деятельности, связанной с разработкой программного обеспечения для решения профессиональных задач, и, является одним из основных предметов, способствующих развитию алгоритмической культуры и компьютерной грамотности студентов.

Курс способствует формированию фундаментальной базы, необходимой для успешного освоения как общепрофессиональных, так и специальных дисциплин, изучение которых связано с созданием информационных систем для различных предметных областей, их анализом, внедрением и сопровождением.

Целью изучения данной дисциплины является ознакомление студентов с понятием алгоритма, способами и средствами их представления, различными парадигмами программирования, классификацией и эволюцией языков программирования, и современными тенденциями их развития, а также детальное изучение одного из языков высокого уровня (язык C++).

2. Место дисциплины в структуре образовательной программы

Данная дисциплина относится к части образовательной программы, формируемой участниками образовательных отношений. Для успешного усвоения данной дисциплины необходимо, чтобы студент овладел также знаниями, умениями и навыками, которые формируются в процессе изучения дисциплин: «Математический анализ», «Алгебра и геометрия», «Дискретная математика» - знать основные понятия и методы математического анализа, алгебры, геометрии; уметь решать задачи теории пределов функций, дифференцирования, интегрирования и разложения функций в ряды; владеть навыками использования стандартных методов и моделей математического анализа и их применения к решению прикладных задач. Знания и навыки, полученные при изучении дисциплины "Практикум по основам программирования", используются учащимися при изучении последующих общепрофессиональных и специальных дисциплин компьютерного цикла, в частности дисциплин "Информатика", "Практикум по объектно-ориентированному программированию", "Языки и методы программирования", а также "Архитектура компьютеров" и "Операционные системы". Знания и практические навыки, полученные в результате освоения дисциплины, используются студентами при разработке курсовых и дипломных работ, в научно-исследовательской работе.

3. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения образовательной программы

Процесс изучения дисциплины направлен на формирование следующих компетенций в соответствии с ФГОС ВО, ООП ВО и приобретения следующих знаний, умений, навыков и (или) опыта деятельности:

| Формируемая компетенция (код и формулировка) | Индикатор достижения компетенции (код и формулировка) | Перечень планируемых результатов обучения |
|-----------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Профессиональные компетенции | | |
| ПК-3 Способен к разработке и применению алгоритмических и программных решений | И-ПК-3.1 Обладает устойчивыми знаниями в области разработки алгоритмов и программирования | Знать: – общие принципы построения и использования современных языков программирования высокого уровня; – базовые структуры данных. |

| | | |
|-------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| в области системного и прикладного программного обеспечения | | <ul style="list-style-type: none"> – основы программирования на языке C++; – принципы разработки программ и отдельных программных модулей. – принципы процедурного и объектно-ориентированного программирования Владеть: <ul style="list-style-type: none"> – навыками алгоритмического мышления. – методиками современной технологии программирования. |
| | И-ПК-3.2 Имеет навыки разработки и реализации алгоритмов в области системного и прикладного программного обеспечения | Знать: <ul style="list-style-type: none"> – средства описания алгоритмов; – различные парадигмы программирования. Уметь: <ul style="list-style-type: none"> – формализовать поставленную задачу; – составлять программы на языке программирования высокого уровня для решения вычислительных задач и задач обработки информации; – проводить оценку сложности алгоритмов; – пользоваться инструментальными средствами для разработки прикладных приложений. - тестировать и отлаживать программы; |
| | И-ПК-3.3 Обладает способностью критического анализа и совершенствования разрабатываемых алгоритмов и программ | Знать: <ul style="list-style-type: none"> – современные технологии программирования Уметь: <ul style="list-style-type: none"> – работать с современными системами программирования, включая объектно-ориентированные; – оформлять программную документацию – оценивать качество готового программного продукта Владеть навыками: <ul style="list-style-type: none"> – разработки алгоритмов решения типовых профессиональных задач; – использования инструментальных средств для создания, отладки и тестирования программ и отдельных модулей, библиотек; – работы с научно-технической литературой и технической документацией по программному обеспечению. |

4. Объем, структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 4 зачетных единиц, 144 акад. часов.

| № п/п | Темы (разделы) дисциплины, их содержание | Семестр | Виды учебных занятий, включая самостоятельную работу студентов, и их трудоемкость (в академических часах) | | Формы текущего контроля успеваемости Форма промежуточной аттестации <i>(по семестрам)</i> |
|-------|------------------------------------------|---------|-----------------------------------------------------------------------------------------------------------|--|-----------------------------------------------------------------------------------------------------|
| | | | Контактная работа | | |

| | | | лекции | практические | лабораторные | консультации | аттестационные испытания | самостоятельная работа | |
|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|---|--------|--------------|--------------|--------------|-----------------------------|---------------------------|---------------------------|
| 1 | Вводная лекция. Алгоритмы и их представление. Языки программирования. Трансляция и выполнение программ | 1 | | | 6 | | | | |
| 2 | Структура программы на языке C++, основные типы данных. Операции, выражения и операторы в языке C++. Простейшие средства ввода и вывода. | 1 | | | 6 | 1 | | 3 | Лабораторная работа № 1 |
| 3 | Условный оператор. Выбор вариантов в языке C++. Програ- мирование циклических процессов. | 1 | | | 6 | 1 | | 3 | Лабораторная работа № 2-3 |
| 4 | Указатели. Работа с динамической памятью. Арифметика указателей. Ссылки. Одномерные массивы: статические, динамические. Контейнер vector(STL) | 1 | | | 6 | 1 | | 3 | Лабораторная работа № 4-5 |
| 5 | Функции, способы передачи параметров. Указатель на функцию. | 1 | | | 8 | 1 | | | |
| 6 | Символьная информация. Средства потокового ввода и вывода. Файлы. | 1 | | | 8 | 1 | | 3 | Лабораторная работа № 6 |
| 7 | Многомерные массивы. Способы представления. | 1 | | | 8 | 1 | | 3 | Лабораторная работа № 7 |
| | | | | | | | 0,3 | 2,7 | зачет |
| | Итого за 1 семестр | | | | 48 | 6 | 0,3 | 17,7 | |
| 8 | Использование структур для определения пользовательских типов данных. | 2 | | | 8 | 1 | | 2 | Лабораторная работа № 8 |
| 9 | Использование библиотеки OpenGL. | 2 | | | 8 | 1 | | 3 | Лабораторная работа № 9 |
| 10 | Жадные алгоритмы. Динамическое программирование. | 2 | | | 8 | 1 | | 2 | Лабораторная работа № 10 |

| | | | | | | | | | |
|----|---------------------------------------------------------------------|---|--|--|-----------|-----------|------------|-------------|--------------------------|
| 11 | Динамические структуры данных. Реализация на базе массива и списка. | 2 | | | 8 | 1 | | 3 | Лабораторная работа № 11 |
| 12 | Основы объектно-ориентированного подхода. | 2 | | | 8 | 1 | | 2 | Лабораторная работа № 12 |
| 13 | Статические и динамические библиотеки | 2 | | | 8 | 1 | | 3 | Лабораторная работа № 13 |
| | | | | | | | 0,3 | 2,7 | зачет |
| | Всего за 2 семестр | | | | 48 | 6 | 0,3 | 17,7 | |
| | ИТОГО | | | | 96 | 12 | 0,6 | 35,4 | |

5. Образовательные технологии, в том числе технологии электронного обучения и дистанционные образовательные технологии, используемые при осуществлении образовательного процесса по дисциплине

В процессе обучения используются следующие образовательные технологии:

Лабораторная работа – организация учебной работы с реальными материальными и информационными объектами, экспериментальная работа с аналоговыми моделями реальных объектов.

Консультации – вид учебных занятий, являющийся одной из форм контроля самостоятельной работы студентов. На консультациях по просьбе студентов рассматриваются наиболее сложные моменты при освоении материала дисциплины, преподаватель отвечает на вопросы студентов, которые возникают у них в процессе самостоятельной работы.

6. Перечень лицензионного и (или) свободно распространяемого программного обеспечения, используемого при осуществлении образовательного процесса по дисциплине

В процессе осуществления образовательного процесса по дисциплине используются:
для формирования материалов для текущего контроля успеваемости и проведения промежуточной аттестации, для формирования методических материалов по дисциплине:

- программы Microsoft Office;
- издательская система LaTeX;
- Adobe Acrobat Reader.

7. Перечень современных профессиональных баз данных и информационных справочных систем, используемых при осуществлении образовательного процесса по дисциплине (при необходимости)

В процессе осуществления образовательного процесса по дисциплине используются:

- Автоматизированная библиотечно-информационная система «БУКИ-NEXT»
http://www.lib.uniyar.ac.ru/opac/bk_cat_find.php
- Электронно-библиотечная система «Юрайт» <https://urait.ru>
- Электронно-библиотечная система «Лань» <http://e.lanbook.com/>
- Электронно-библиотечная система «Консультант Студента»:
<https://www.studentlibrary.ru/>

8. Перечень основной и дополнительной учебной литературы, ресурсов информационно-телекоммуникационной сети «Интернет» (при необходимости), рекомендуемых для освоения дисциплины

а) основная литература

1. О. В. Власова, Н. П. Федотова, О. П. Якимова Основы программирования: учебное пособие - Ярославль: ЯрГУ, 2019 <http://www.lib.uniyar.ac.ru/edocs/iuni/20190204.pdf>
2. Подбельский, В. В. Язык Си++ : учеб. пособие / В. В. Подбельский. - 5-е изд. - Москва : Финансы и статистика, 2022. - 560 с. - ISBN 978-5-00184-082-4. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : <https://www.studentlibrary.ru/book/ISBN9785001840824.html>
3. М. В. Огнева, Е. В. Кудрина Программирование на языке C++: практический курс: учебное пособие для вузов — Москва : Издательство Юрайт, 2022 <https://urait.ru/viewer/programmirovanie-na-yazyke-s-prakticheskiy-kurs-492984>

б) дополнительная литература

1. В. В. Подбельский, С. С. Фомин Курс программирования на языке Си: учебник - М., ДМК Пресс, 2018. <https://www.studentlibrary.ru/ru/doc/ISBN9785940749479-SCN0000/000.html>
2. Павловская Т. А. C/C++. Программирование на языке высокого уровня: учебник для вузов - СПб.: Питер, 2010.
3. Страуструп, Б. Язык программирования C++ для профессионалов / Страуструп Б. - Москва : Национальный Открытый Университет "ИНТУИТ", 2016. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/intuit_418.html
4. Кувшинов Д. Р. Основы программирования: учебное пособие для вузов — Москва: Издательство Юрайт, 2022. <https://urait.ru/viewer/osnovy-programmirovaniya-493460>
5. Лафоре Р. Объектно-ориентированное программирование в C++. - СПб.: Питер, 2016.
6. Кнут Д. Искусство программирования ЭВМ. Т.1, Основные алгоритмы — М.: Мир, 1976.
7. Вирт Н. Алгоритмы и структуры данных: Пер. с англ. - М.: Мир, 1989
8. О. В. Власова, Н. П. Федотова, О. П. Якимова Основы программирования: учебное пособие. - Ярославль: ЯрГУ, 2021.
9. С. Г. Волченков, П. А. Корнилов, Ю. А. Белов, Н. Л. Дашниц, В. А. Никулин, Н. И. Заводчикова Ярославские олимпиады по информатике: сборник задач с решениями - М., БИНОМ. Лаборатория знаний, 2010.

9. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине включает в свой состав специальные помещения:

- учебные аудитории для проведения групповых и индивидуальных консультаций,
- учебные аудитории для проведения текущего контроля и промежуточной аттестации;
- помещения для самостоятельной работы;
- помещения для хранения и профилактического обслуживания технических средств обучения.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа к электронной информационно-образовательной среде ЯрГУ.

Автор(ы):

Старший преподаватель кафедры компьютерной
безопасности и математических методов обработки информации

О.В. Власова

**Приложение № 1 к рабочей программе дисциплины
«Практикум по основам программирования»**

**Фонд оценочных средств
для проведения текущего контроля успеваемости
и промежуточной аттестации студентов
по дисциплине**

**1. Типовые контрольные задания и иные материалы,
используемые в процессе текущего контроля успеваемости**

Лабораторные работы

Лабораторная работа №1 Типы данных. Условный оператор.

Лабораторная работа состоит из четырех заданий.

Задание 1 (И-ПК-3.1)

Составить программу для Машины Тьюринга.

Для сокращения формулировки задач введём следующие два соглашения:

- буквой Р будем обозначать входное слово;
- буквой А будем обозначать алфавит входного слова, т.е. набор тех символов, из которых и только которых может состоять Р (отметим, однако, что в промежуточных и выходном словах могут появляться и другие символы).

$A=\{0,1,2,3\}$. Считая непустое слово Р записью числа в четверичной системе счисления, получить запись этого числа в двоичной системе.

Задание 2

Работа с вещественными числами. Математические функции. (И-ПК-3.2)

Для выполнения задания необходимо знать:

- диапазоны различных вещественных типов;
- описание и инициализацию переменных вещественных типов;
- форматы представления вещественных чисел: с фиксированной и с плавающей точкой;
- операторы ввода-вывода вещественных чисел в нужном формате, с нужной точностью;
- арифметические операции над переменными вещественных типов;
- принципы сравнения вещественных чисел с учетом погрешности вычислений;
- различные функции для округления вещественных чисел;
- основные математические функции, определенные в библиотеке <cmath>.

В этом задании необходимо ввести значения необходимых переменных и вычислить два выражения А и В, зависящих от этих переменных, вывести значения обоих выражений и разности С между ними в формате с фиксированной точкой с точностью 0,000001.

Выражения могут содержать дроби и корни, поэтому важно помнить об ОДЗ. Поскольку оператор ветвления в этой лабораторной работе не используется, необходимо вывести информацию об ОДЗ на экран.

Следует также помнить о читаемости кода и не записывать вычисление выражения в один оператор.

При составлении системы тестов к заданию может потребоваться ввести углы в градусах, а потом перевести их в радианы.

$$1. A = \left(\frac{4 - 2x + x^2}{4 - 2x} + \frac{6x^2 + 8 + 12x}{4 - x^2} - \frac{x^2 + 2x + 4}{2x + 4} \right)^{-\frac{1}{3}} \cdot (x + 2);$$

$$B = \sqrt[3]{4 - x^2};$$

Задание 3. (ИД-ПК-3.3)

Решение уравнений.

Для выполнения задания необходимо знать:

- ввод, вывод, инициализацию целых и вещественных переменных;
- правила преобразования переменных различного типа;
- стандартные логические функции: отрицание, конъюнкция, дизъюнкция;
- приоритет логических операций.

Решение уравнений

Необходимо аккуратно рассмотреть все возможные принципиально разные варианты значений коэффициентов или параметров.

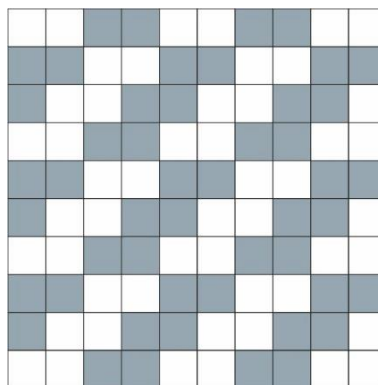
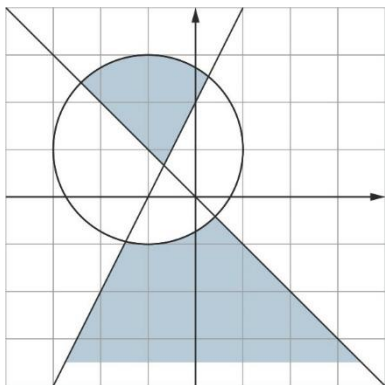
Даны два целых числа: a и b . Решите уравнение $ax + b = 0$.

Необходимо вывести все решения, если их число конечно; NO, если решений нет; и INF, если решений бесконечно много.

Задание 4 (ИД-ПК-3.2)

Проверка принадлежности точки области

Это задание состоит из двух пунктов. В обоих случаях нужно проверить, принадлежит ли точка закрашенной области на плоскости. Точка задается двумя вещественными координатами. В первом случае - любыми, во втором пункте - неотрицательными координатами. Во втором пункте считаем началом координат левый нижний угол картинки. В заданиях второго пункта, если область ограничена не прямой линией, то это дуга окружности. Часто на рисунке изображена не вся область, а только ее часть, достаточная для обобщения на всю плоскость. Масштаб: 1 - одна клеточка по обеим осям.



Лабораторная работа №2 Цикл for. Перебор. Ребусы

Лабораторная работа состоит из трех заданий.

Задание 1. (ИД-ПК-3.1)

В этом задании необходимо написать программу, которая перебором вариантов решает числовой ребус.

В ребусе разным буквам соответствуют разные цифры, и наоборот. Числа не могут начинаться с нуля. Решений может быть несколько, тогда нужно вывести их все.

В некоторых случаях можно ограничить перебор вариантов (когда цифра может быть только четной или заведомо не превосходит какого-то числа).

$$\text{ДА} + \text{ДА} = \text{НЕТ}$$

Задание 2 (ИД-ПК-3.2)

Цикл for. Печать таблицы истинности булевой функции (И-ОПК-1_1)

Для вычисления логических выражений потребуются логические функции от двух переменных. К основным будем относить:

отрицание (не,!);

конъюнкция (и, &,*) логическое умножение в формуле может быть опущено;

дизъюнкция (или, +)

Другие логические функции можно выразить через основные. Для каждой из них необходимо реализовать отдельный метод вычисления.

- $x \oplus y = \bar{x}y + x\bar{y}$;

- $x \rightarrow y = \bar{x} + y$;

- $x \leftrightarrow y = xy + \bar{x}\bar{y}$;

- $x \mid y = \overline{\bar{x}\bar{y}}$;

- $x \downarrow y = \overline{x + y}$;

Определим следующий приоритет логических операций:

\neg , \wedge , $\vee(+)$, \oplus , \rightarrow , \leftrightarrow , \mid , \downarrow .

В этом задании необходимо построить таблицы истинности для двух логических выражений A и B.

1. $A = \bar{x}\bar{z} + xy + x\bar{z}$; $B = z \rightarrow xy$;

Задание 3.

Цикл while. Сумма ряда (ИД-ПК-3.3)

Дана функция, заданная бесконечным рядом (по вариантам).

Ее нужно протабулировать в N точках отрезка [A, B]. Если некоторые точки отрезка не принадлежит указанной в задании области сходимости ряда, необходимо вывести сообщение об этом. Значения функции вычисляются с заданной точностью epsilon.

Таким образом на вход поступают 4 числа: N, A, B, epsilon.

Функцию нужно вычислить тремя способами, а результат вывести на экран в виде таблицы:

| x | f1(x) | f2(x) | f3(x) |
|-------|-------|-------|-------|
| 0.753 | 0.753 | 0.753 | 0.753 |
| | | | |

Количество знаков после запятой должно соответствовать заданной точности.

Способы вычисления:

1. вызвать стандартную функцию, определенную в пространстве имен Math;
2. вычисление каждого слагаемого по-отдельности, используя самостоятельно написанные для этого функции возведения в степень, вычисления модуля, факториала и, возможно, других функций;
3. слагаемые надо вычислять эффективно, используя результаты, полученные на предыдущем шаге. Без использования вспомогательных функций (за исключением вариантов с числами Бернулли).

Сами функции $f_i(x)$ необходимо реализовать в виде отдельных методов `double fi(double x, double e)`.

$$1. f(x) = \sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}, x \in R$$

Лабораторная работа №3 Циклы. Последовательности.

Лабораторная работа состоит из четырех заданий

Задание 1.

Обработка последовательностей (ИД-ПК-3.1)

Цель работы. Придумать алгоритм и решить задачу на применение простейших конструкций языка - цикла и условия, осуществляя пошаговый ввод и вывод данных (массив использовать нельзя).

Требуется ввести с клавиатуры натуральное число n , далее в цикле обеспечить ввод и обработку остальных данных.

Даны натуральные числа n, a_1, \dots, a_n . Определить количество членов a_i последовательности a_1, \dots, a_n кратных 3 и не кратных 5.

Задание 2. Обработка последовательностей (ИД-ПК-3.2)

Цель работы. Придумать алгоритм и решить задачу на применение простейших конструкций языка - цикла и условия, осуществляя пошаговый ввод и вывод данных.

Решение задачи должно быть эффективным как по времени работы, так и по используемой памяти. Программу будем считать эффективной по памяти, если используемая память не зависит от размера входных данных. Программу будем считать эффективной по времени, если при увеличении размера входных данных N в t раз (t - любое число) время её работы увеличивается не более чем в t раз.

Дан список точек плоскости с целочисленными координатами. Необходимо определить:

- 1) номер координатной четверти K , в которой находится больше всего точек;
- 2) точку A в этой четверти, наименее удаленную от осей координат;
- 3) расстояние R от этой точки до ближайшей оси.

Если в нескольких четвертях расположено одинаковое количество точек, следует выбрать ту четверть, в которой величина R меньше. При равенстве и количества точек, и величины R необходимо выбрать четверть с меньшим номером K . Если в выбранной четверти несколько точек находятся на одинаковом минимальном расстоянии от осей координат, нужно выбрать первую по списку. Точки, хотя бы одна из координат которых равна нулю, считаются не принадлежащими ни одной четверти и не рассматриваются. Напишите эффективную, в том числе по памяти, программу, которая будет решать эту задачу.

Описание входных данных

В первой строке вводится одно целое положительное число – количество точек N . Каждая из следующих N строк содержит координаты очередной точки – два целых числа (первое – координата x , второе – координата y).

Описание выходных данных

Программа должна вывести номер выбранной четверти K , количество точек в ней M , координаты выбранной точки A и минимальное расстояние R по образцу, приведённому ниже в примере.

Пример входных данных:

```
7
-3 4
1 2
1 1
0 4
```

-2 -3

-6 8

-12 1

Пример выходных данных для приведённого выше примера входных данных:

$K = 2$

$M = 3$

$A = (-12, 1)$

$R = 1$

Лабораторная работа №4 Одномерные массивы.

Лабораторная работа состоит из трех заданий.

Задание 1.

Обработка одномерных массивов. (ИД-ПК-3.1)

Цель работы. Решить задачу на обработку одномерного массива. Требуется ввести с клавиатуры размерность массива n , далее в цикле ввести элементы массива. Полученный ответ вывести на экран.

Варианты задания

Даны натуральные n, m , целые числа a_1, \dots, a_n и b_1, \dots, b_m . Внутри каждой из данных последовательностей нет повторяющихся членов. Построить объединение данных последовательностей.

Задание 2.

Обработка серий подряд идущих элементов. (ИД-ПК-3.2)

Цель работы. Решить задачу на обработку одномерного массива. Требуется ввести с клавиатуры размерность массива n , далее в цикле ввести элементы массива. Полученный ответ вывести на экран. Если в результате работы размер Вашего массива может измениться в сторону увеличения, то следует заранее это учесть, создав массив большего размера.

Определение: Назовем серией группу подряд идущих одинаковых элементов, а длиной серии — количество этих элементов (длина серии может быть равна 1).

Дан целочисленный массив A размера N . Сформировать два новых целочисленных массива B и C одинакового размера, записав в массив B длины всех серий исходного массива, а в массив C — значения элементов, образующих эти серии.

Задание 3.

Методы, принимающие и возвращающие массивы в качестве параметров. Файлы. (ИД-ПК-3.3)

При выполнении этого задания нужно использовать одномерные массивы и написать методы, принимающие и возвращающие массивы в качестве параметров. Кроме перечисленных в заданиях методов необходимо реализовать методы ввода и вывода массивов. Не следует объединять в один метод реализацию нескольких задач. Массивы и их длины в условии задачи записаны заглавными буквами для лучшей читаемости условия. Если в результате работы размер Вашего массива может измениться в сторону увеличения, то следует заранее это учесть, создав массив большего размера. Исходные данные считываются из файла. Результат записывается в файл.

В соответствии с вариантом Вам необходимо выполнить следующее:

Из целочисленного массива $X(N)$ все четные элементы записать в массив $Y(K)$. Удалить в массивах максимальные элементы (их может быть несколько). В программе написать методы: формирования массива Y , поиска максимального элемента, удаления элемента, удаления максимальных элементов

Лабораторная работа №5 Датчик случайных чисел. Файлы. Контейнер vector

Лабораторная работа состоит из двух заданий.

Задание 1.

Датчик случайных чисел. (ИД-ПК-3.3)

Цель работы. Научиться формировать с помощью датчика случайных чисел наборы тестовых данных для проверки корректности работы программы.

Пользователь задает имя тестового набора (NameSet, например test), количество файлов в наборе (N), тип данных в наборе (int, double), диапазон значений элементов в наборе (Min, Max), максимальное и минимальное количество элементов в файле (LenMin, LenMax), допустимость повторения элементов (yes, no), необходимость сортировки (asc, desc, none), тип вывода элементов (row, col).

Вам необходимо написать метод, который по заданным параметрам сформирует набор тестовых файлов.

А затем на сформированном наборе тестов проверить работу Вашего кода из заданий 1, 2 и 3.

Пример 1:

NameSet = 'test', N=2, Type = 'int' , Min = 2, Max = 10, LenMin=4, LenMax = 10, Repeat = 'yes' , Sort = 'asc', TypeOut = 'row'

Может быть сформирован набор из 2-х файлов

test01

4

2 4 7 9

test02

8

3 3 3 6 6 7 8 9

Пример 2:

NameSet = 'input', N=3, Type = 'double' , Min = 0, Max = 1, LenMin=2, LenMax = 10, Repeat = 'no' , Sort = 'none', TypeOut = 'col'

Может быть сформирован набор из 3-х файлов

input01

3

0,24

0,01

0,0123

input02

2

1

0

input03

4

0,1

0,2

0,3

0,4

Задание 2.

Работа с контейнером vector (ИД-ПК-3.2)

В проекте «СкупойПлатитДважды» 1 января решено тратить на развитие 60% накоплений всех участников. При этом 20% самых богатых участников вносят 80% от своих накоплений, остальные участники вносят равный процент таким образом, чтобы общая сумма взносов всех участников составила 60%, обозначенные выше.

Запишите в ответе два целых числа: сумма взноса от всех «богатых» участников проекта и сумма взноса участника с самым небольшим размером накоплений. Если в результате получаются дробные числа, нужно записать их целые части.

Входные данные. В первой строке натуральное число N – количество участников проекта ($20 \leq N \leq 1000$). В следующих N строках находятся значения – размер накоплений всех пользователей (все числа натуральные, не превышающие 1000), каждое в отдельной строке.

Пример входного файла:

10
10
12
25
25
40
35
18
19
10
12

При таких исходных данных ответ должен содержать 2 числа – 60 и 4.

Примечание: если при нахождении 20% от количества участников получается нецелое число, нужно взять его целую часть.

Лабораторная работа №6 Обработка строк

Лабораторная работа состоит из двух заданий.

Задание 1. (ИД-ПК-3.2)

Цель работы. Решить задачу на обработку текста, используя функции работы со строками. Всюду ниже, если это не оговорено особо, предполагается, что исходным является текстовый файл. Если не оговорено условием, то в текст могут входить слова из латинских букв, цифры, знаки арифметических операций, точка, запятая, пробел.

Требуется считать текст из файла, вывести его на экран, после решения задачи вывести на экран результат. Решение должно быть эффективным!!!

Словом, называется группа букв, за которой и перед которой стоит не буква, а другой символ – цифра, знак, пробел и т.п.

Например, в строке «abc2a, to be, 3red... » ПЯТЬ слов: abc, a, to, be, red.

Варианты.

1. Текстовый файл содержит строку из десятичных цифр, всего не более чем из 10^6 символов. Файл образовался в результате последовательной записи «таймкодов» некоторых событий в формате ННММ (часы и минуты слитно по две цифры, т.е. всего 4 цифры на «таймкод», от 0000 до 2359) и прочих случайных данных. Найдите максимально возможное количество подряд идущих «таймкодов» между фрагментами случайной информации. Например, в строке 4212231135414447 можно выделить таймкоды тремя способами: 4[2122]3[1135]4[1444]7, 42[1223,1135]4[1444]7 или 421[2231,1354,1444]7. В последнем случае получилось наибольшее количество таймкодов подряд (3), это число и нужно ввести в ответе.

Задание 2. (ИД-ПК-3.3)

Цель работы. Реализовать шифр простой замены по заданному алгоритму.

Шифр простой замены — метод шифрования, который заключается в создании таблицы шифрования по определённому алгоритму. В такой таблице каждой букве открытого текста однозначно соответствует буква шифр-текста. Шифрование заключается в замене букв открытого текста согласно таблице. Для расшифровки нужно знать таблицу или алгоритм, по которому она создается. В задании описан алгоритм создания такой таблицы.

В данном задании необходимо выполнить следующее:

Зашифровать оригинальный текст с помощью алгоритма простой замены (по вариантам)

1. Для работы желательно взять осмысленный текст (любимую книгу или отрывок из нее), содержащий не менее 10 страниц,
2. текст нужно прочитать из файла,
3. зашифрованный текст записать в файл.

Расшифровать свой текст, убедиться, что дешифрованный текст совпадает с исходным.

Для того, чтобы облегчить дальнейшую расшифровку текста в задании 2, будем придерживаться следующих соглашений:

- кодируем шифром простой замены только буквы, все остальные символы
- оставляем как есть;
- букву Ё не кодируем, оставляем, как есть;
- регистр символов сохраняем;
- используем предварительно обработанный текст узнаваемого с помощью поисковых систем художественного произведения (или его значительной части).

Варианты.

1. Шифр Гая Юлия Цезаря. Циклический сдвиг алфавита влево на 3 позиции.

Лабораторная работа №7 Обработка двумерных массивов

Лабораторная работа состоит из двух заданий.

Задача 6а. (ИД-ПК-3.2)

Цель работы. Использование подпрограмм в задаче на обработку двумерного массива.

Требуется вывести на экран меню, состоящее из следующих пунктов:

1. ввод матрицы с клавиатуры,
2. ввод матрицы из файла,
3. вычисление характеристики,
4. преобразование матрицы,
5. печать матрицы,
6. выход.

и обеспечить его функционирование.

Внутри программы характеристика оформляется в виде метода с передачей параметров по значению. Этот метод возвращает значение булевского типа; преобразование в виде метода с передачей параметров по ссылке. Необходимо отслеживать, был ли произведен ввод данных до выбора пунктов меню, которые обрабатывают матрицу.

Варианты характеристика/ преобразование.

Характеристика. В матрице существует строка, элементы которой образуют симметричную последовательность.

Преобразование. Удалить из матрицы строку и столбец, на пересечении которых расположен максимальный по модулю элемент (если таких элементов несколько, то максимальный элемент с меньшими коэффициентами).

Задача 6б. (ИД-ПК-3.2)

В таблице из n строк и n столбцов некоторые клетки заняты шариками, другие свободны. Выбран шарик, который нужно переместить, и место, куда его нужно переместить. Выбранный шарик за один шаг перемещается в соседнюю по горизонтали или вертикали свободную клетку. Требуется выяснить, возможно ли переместить шарик из начальной клетки в заданную, и если возможно, то найти путь из наименьшего количества шагов.

Лабораторная работа №8 Структуры. Многофайловый проект. (ИД-ПК-3.2)

Цель работы. Разработать структуру (класс) по описанию варианта. Для структуры (класса) определить методы: конструктор, вывод данных, сравнение. Исходные данные на выбор пользователя (клавиатура или файл). Для хранения данных использовать вектор. Печать вектора, сортировку и поиск элементов в векторе оформить в виде функций.

Вариант 1

Описать структуру (класс) с именем **STUDENT**, содержащую следующие поля: номер(int); фамилия и инициалы(char *, string); номер группы (аббревиатура специальности, курс) (char *, string); успеваемость (массив из пяти элементов(int)).

Написать программу, выполняющую следующие действия:

- Ввод данных (файл, клавиатура на выбор пользователя).
 - Упорядочить записи по возрастанию номера группы.
 - Вывод на экран фамилий и номеров групп для всех студентов, включенных в список, если средний балл студента больше K (float).
 - Вывод на экран фамилии и успеваемости для всех студентов, включенных в список, если группа равна N (char *, string).
- Если информация не найдена, вывести соответствующее сообщение.

Предусмотреть возможность редактирования элементов списка – удаление, добавление, изменение.

Лабораторная работа №9 Геометрия+OpenGL. (ИД-ПК-3.3)

Цель работы. Решить задачу и отобразить решение графически на экране.

Исходные данные прочитать из текстового файла. Решение должно быть эффективным!!! Используйте структуры для представления геометрических объектов.

Задача решается в 2 этапа.

1 этап: Построение математической модели задачи, разработка алгоритма решения, тестирование.

2 этап: Отрисовка найденного решения на экране с использованием библиотеки OpenGL.

Задание

Многоугольник на плоскости задается координатами своих N вершин в порядке обхода их по контуру по часовой стрелке (контур самопересечений не имеет). Для заданной точки $Z(x, y)$ определить, принадлежит ли она стороне многоугольника или лежит внутри, или вне его.

Лабораторная работа №10 Жадные алгоритмы. Динамическое программирование

Работа состоит из трех заданий а, б и с.

а- жадный алгоритм

б и с - динамическое программирование.

Перед кодом программы размещается текст задания(комментарий). Затем приводится набор тестов. Данные для тестов размещаются в отдельных файлах(*.txt). Код программы должен содержать комментарии, поясняющие основные части алгоритма. Код должен быть структурирован.

Задание а. (ИД-ПК-3.2)

Цель работы. Решить задачу с использованием жадного алгоритма.

Варианты:

Дана лекционная аудитория, в которой несколько профессоров хотят прочесть свои лекции. Для составления расписания профессора подали заявки, вида $[si;fi]$ – время начала и конца лекции. Лекция считается открытым интервалом, то есть какая-то лекция может начаться в момент окончания другой, без перерыва. Составьте расписание занятий так, чтобы выполнить максимальное количество заявок.

Входные данные

В первой строке вводится натуральное число N , не более 1000 – общее количество заявок. Затем вводится N строк с описаниями заявок - по два числа в каждом si и fi .

Гарантируется, что $si < fi$. Время начала и окончания лекции – натуральное число, не превышает 1440 (в минутах с начала суток :))

Выходные данные

Выведите одно число – максимальное количество заявок, которые можно выполнить.

Пояснения к примерам

Во втором примере можно выполнить вторую и третью заявки.

Примеры

входные данные

1
5 10

выходные данные

1

входные данные

3
1 5
2 3
3 4

выходные данные

2

Задание б. (ИД-ПК-3.2)

При переработке радиоактивных материалов образуются отходы трех видов — особо опасные (тип А), неопасные (тип В) и совсем не опасные (тип С). Для их хранения используются одинаковые контейнеры. После помещения отходов в контейнеры последние укладываются вертикальной стопкой. Стопка считается взрывоопасной, если в ней подряд идет более одного контейнера типа А. Стопка считается безопасной, если она не является взрывоопасной. Для заданного количества контейнеров N определить число безопасных стопок.

Входные данные

Одно число $1 \leq N \leq 20$.

Выходные данные

Одно число — количество безопасных вариантов формирования стопки.

Примечание

В примере из условия среди стопок длины 2 бывают безопасные стопки типов АВ, АС, ВА, ВВ, ВС, СА, СВ и СС. Стопки типа АА являются взрывоопасными.

Примеры

входные данные

2

выходные данные

8

Задание с. (ИД-ПК-3.3)

С севера на юг 2

Исходные данные для Робота записаны электронной таблицы прямоугольной формы. Роботу нужно перейти через поле с севера (верхняя строка) на юг (нижняя строка). Он может начать переход с любой клетки верхней строки и закончить на любой клетке нижней строки. С каждым шагом Робот переходит в следующий ряд и может за одно перемещение попасть в одну из трех клеток следующей строки (на клетку прямо или боковые с ней). Ходы только вбок (без смены строки) и/или назад запрещены. В каждой клетке поля лежит монета достоинством от 1 до 100. Робот собирает все монеты по пройденному маршруту.

Известно, что Робот собрал максимальное количество монет, пройдя с северной границы поля (сверху) до южной границы поля (снизу). В ответе укажите два числа – количество монет из первой и последней клетки маршрута.

Лабораторная работа №11 Структуры данных.**Задание 1. (ИД-ПК-3.2)**

Цель работы. Выполнить реализацию шаблонного однонаправленного или двунаправленного списка в виде класса. Определить для списка необходимые методы: Конструктор, Добавление элемента в список в сортированном порядке; Поиск элемента по критерию; Удаление элемента; Редактирование элемента; Печать всего списка; Печать элементов списка, удовлетворяющих критерию поиска; Деструктор.

Описать заданный класс. Для класса определить методы: конструктор (по умолчанию, с параметрами), оператор вывода, оператор ввода, оператор сравнения, геттеры и сеттеры, если они необходимы. Исходные данные прочитать из текстового файла или с консоли на выбор пользователя.

Четные номера – однонаправленный список. Нечетные номера – двунаправленный список.

Вариант 1

Описать класс с именем **STUDENT**, содержащий следующие поля: номер; фамилия и инициалы; номер группы; успеваемость (массив из пяти элементов). Написать программу, выполняющую следующие действия:

- Построение списка, ввод данных (файл, клавиатура на выбор пользователя); записи в списке должны быть упорядочены по возрастанию номера группы.
- Вывод на экран фамилий и номеров групп для всех студентов, включенных в список, если средний балл студента больше K(float).
- Вывод на экран фамилии и успеваемость для всех студентов, включенных в список, если группа равна N(char *, string).

Если нет таких студентов, вывести соответствующее сообщение.

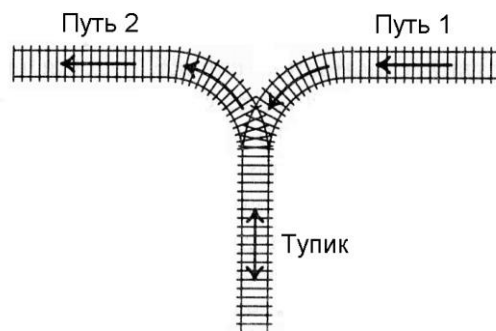
Предусмотреть возможность редактирования элементов списка – удаление, добавление, изменение.

Задание 2. (ИД-ПК-3.3)

Структуры данных. Списки, стеки, очереди.

Цель работы. В работе необходимо использовать динамические структуры данных. Выбор конкретной структуры (список, стек и т. д.) определяется удобством использования, если в самом задании явно не указан конкретный вид структуры.

К тупику со стороны пути 1 (см. рисунок) подъехал поезд. Разрешается отцепить от поезда один или сразу несколько первых вагонов и завезти их в тупик (при желании, можно даже завезти в тупик сразу весь поезд). После этого часть из этих вагонов вывезти в сторону пути 2. После этого можно завезти в тупик еще несколько вагонов и снова часть оказавшихся вагонов вывезти в сторону пути 2. И так далее (так, что каждый вагон может лишь один раз заехать с пути 1 в тупик, а затем один раз выехать из тупика на путь 2). Заезжать в тупик с пути 2 или выезжать из тупика на путь 1 запрещается. Нельзя с пути 1 попасть на путь 2, не заезжая в тупик.



Известно, в каком порядке изначально идут вагоны поезда. Требуется с помощью указанных операций сделать так, чтобы вагоны поезда шли по порядку (сначала первый, потом второй и т.д., считая от головы поезда, едущего по пути 2 в сторону от тупика).

Формат входных данных

Вводится число N — количество вагонов в поезде ($1 \leq N \leq 2000$). Далее идут номера вагонов в порядке от головы поезда, едущего по пути 1 в сторону тупика. Вагоны пронумерованы натуральными числами от 1 до N , каждое из которых встречается ровно один раз.

Формат выходных данных

Если сделать так, чтобы вагоны шли в порядке от 1 до N , считая от головы поезда, когда поезд поедет по пути 2 из тупика, можно, выведите действия, которые нужно проделать с поездом. Каждое действие описывается двумя числами: типом и количеством вагонов:

- если нужно завезти с пути 1 в тупик K вагонов, должно быть выведено сначала число 1, а затем — число K ($K \geq 1$),
- если нужно вывезти из тупика на путь 2 K вагонов, должно быть выведено сначала число 2, а затем — число K ($K \geq 1$).

Если возможно несколько последовательностей действий, приводящих к нужному результату, выведите любую из них.

Если выстроить вагоны по порядку невозможно, выведите одно число 0.

Примеры

| b.in | b.out |
|-------------|--------------|
| 3 | 1 3 |
| 3 2 1 | 2 3 |
| 4 | 1 2 |
| 4 1 3 2 | 2 1 |
| | 1 2 |
| | 2 3 |
| 3 | 0 |
| 2. 3 1 | |

Лабораторная работа №12 Бинарные деревья

Задание 1 (ИД-ПК-3.1)

Написать программу, обеспечивающую работу со сбалансированным деревом: создание нового дерева из N элементов, добавление/удаление элемента, просмотр дерева, очистка дерева, подсчет глубины.

Деревья общего вида

Дерево общего вида (каждая вершина которого может иметь произвольное число дочерних вершин, расположенных в фиксированном порядке в направлении слева направо) реализуется с помощью набора связанных записей типа TNode следующим образом: для каждой внутренней вершины ее поле Left содержит указатель на ее первую (т. е. левую) дочернюю вершину, а поле Right — указатель на его правого брата, т. е. вершину, имеющую в дереве общего вида того же родителя. Поле Right корня дерева общего вида всегда равно NULL, так как корень братьев не имеет.

Способ задания дерева общего вида:

Дана строка S, содержащая описание непустого дерева общего вида в следующем формате:

| | | |
|----------------------|-----|------------------------------------------------|
| <дерево> | ::= | <вершина> <вершина>(<список поддеревьев>) |
| <список поддеревьев> | ::= | <дерево> <дерево>,<список поддеревьев> |
| <вершина> | ::= | <целое число> |

Например, «3(2,7(6,4,5),8(4(2,3),5(1)))» (пробелы отсутствуют, вершины-сестры перечисляются в порядке слева направо). Создать дерево общего вида по описанию, приведенному в S, и вывести указатель на его корень.

Задание 2 (ИД-ПК-3.2)

Дана строка, описывающая дерево общего вида. Любая вершина исходного дерева имеет не более двух дочерних вершин. Построить дерево по строке. Создать бинарное дерево, соответствующее исходному дереву общего вида, и выполнить его обход в ширину. Считать, что первая дочерняя вершина любой вершины дерева общего вида соответствует левой дочерней вершине в бинарном дереве.

Лабораторная работа №13 ООП. Библиотеки (ИД-ПК-3.1)

I. В соответствии с вариантом задания разработать класс и программу, иллюстрирующую его возможности (меню для демонстрации всех возможностей).

Требования к классу:

- обязательно **наличие закрытой (private) и открытой (public) частей;**
- обязательно **наличие поля типа указатель** в закрытой части.
- класс должен иметь, **по крайней мере, три** конструктора, определенных программистом: **конструктор по умолчанию, конструктор с параметрами (имеет список инициализации) и конструктор копирования;**
- необходимо задать **набор методов для получения значений и модификации** полей данных, находящихся в закрытой части класса(Set, Get);
- для разработанного класса должны быть **перегружены операции:** арифметическая, сравнения(для числа и строки), присваивания и т.п.. Выбор перегружаемых операций определяется семантикой предметной области.
- для разработанного класса должны быть перегружены **операция вывода (дружественная функция).**
- класс должен иметь **деструктор.**

Для класса определить заголовочный файл и файл реализации.

Требование к программе:

1. Построить массив на основе разработанного класса. Заполнение массива реализовать двумя способами: чтение из файла и ввод с клавиатуры.
2. Добавить *обработку исключений* «файл не найден», «ошибка чтения данных из файла», «некорректные данные».
3. Отсортировать массив, используя перегруженные операции сравнения Вашего класса и функцию-шаблон сортировки. Результат сортировки записать в файл. (Формат входного и выходного файла должны совпадать.)

Варианты заданий

1. Политическая партия (Название, Адрес, Руководитель, Количественный состав, Количество голосов собранных в поддержку и т.д.). Возможные операции для перегрузки: сравнение (Количество голосов собранных в поддержку), + (Увеличение количества голосов собранных в поддержку).

Для класса, разработанного в лабораторной работе № 13 построить *статическую и динамическую библиотеки*.

Разработать приложения, использующие эти библиотеки.

2. Список вопросов и (или) заданий для проведения промежуточной аттестации

1 семестр

1. Алгоритм. Свойства. Способы записи.
2. Машина Тьюринга.
3. Архитектура компьютера.
4. Структура программы. Заголовочные файлы, файлы реализации. Препроцессор.
5. Понятие переменной. Типы данных. Области видимости. Время жизни.
6. Арифметические, логические операции. Приоритет.
7. Арифметика указателей. Связь между указателями и массивами. Ссылки.
8. Поточковый ввод-вывод (iostream). Использование манипуляторов.
9. Условный оператор. Оператор выбора.
10. Виды циклов.
11. Псевдослучайные числа. Получение целого(вещественного) числа из заданного отрезка [a, b]
12. Одномерные массивы. Статическая память. Динамическая память.
13. Сортировка выбором. Трудоемкость.
14. Сортировка обменом. Модификации. Трудоемкость.
15. Сортировка вставками. Модификации. Трудоемкость.
16. Быстрая сортировка. Трудоемкость.
17. Сортировка слиянием. Трудоемкость.
18. Задачи поиска минимума (максимума) и его номера в одномерном массиве.
19. Поиск в одномерном массиве(Линейный). Трудоемкость.
20. Бинарный поиск. Бинарный поиск по ответу.
21. Изменение массива (добавление, удаление элементов). Трудоемкость.
22. Статические двумерные массивы.
23. Динамические двумерные массивы
24. Функции. Способы передачи параметров.
25. Рекурсивные функции. Алгоритм Евклида (НОД).

26. Работа с файлами (fstream).
27. Функции работы со строками. Библиотека string.h
28. Функции работы со строками. Библиотека string

Примерный билет к зачету.

Найдите ошибку.

Требовалось написать программу, при выполнении которой с клавиатуры считывается натуральное число N , не превосходящее 10^9 , и выводится произведение цифр этого числа. Стажёр торопился и написал программу неправильно

```
#include <iostream>
using namespace std;
int main(){
    long int N, product;
    int digit;
    cin >> N;
    product = N % 10;
    while (N >= 10){
        digit = N % 10;
        product = product*digit;
        N = N / 10;
    }
    cout << product;
}
```

Последовательно выполните следующее.

1. Что выведет программа при вводе числа 438?
2. Приведите пример числа, такого что несмотря на ошибки программа выдаёт верный результат.
3. Найдите в программе все ошибки (известно, что их не более двух) и исправьте их. Для каждой ошибки выпишите строку, в которой она допущена, и приведите эту же строку в исправленном виде.

Что делает данная программа?

```
#include <iostream>
using namespace std;
int main() {
    string s;
    int i, n, p = 0;
    cin >> s;
    n = s.length();
    for(i=0; i<n; i++){
        if(s[i] == 'z'){p = 1;}
    }
    if(p==1){cout << "YES";} else {cout << "NO";}
    return 0;
}
```

Задание на написание кода

Написать функцию, принимающую на вход произвольное целочисленное значение и возвращающую строку, содержащую его представление в двоичной системе исчисления.

Декларация.

`void NumberAsBinary(char* _result, // строка с результатом`

Приложение № 2 к рабочей программе дисциплины «Практикум по основам программирования»

Методические указания для студентов по освоению дисциплины

Основной формой изложения учебного материала по дисциплине «Практикум по основам программирования» являются лекции дисциплины «Основы программирования», причем в форме лекции-беседы или мастер-класса. Это связано с тем, что для формирования алгоритмического мышления необходим опыт формализации задач и их поэтапного решения. Лабораторные занятия «Практикум по основам программирования» нацелены на закрепление лекционного материала путем применения его к конкретным задачам и отработка навыков работы по разработке алгоритмов, кодированию информации или работы с различным программным обеспечением.

Для успешного освоения дисциплины очень важно решение достаточно большого количества задач, требующих разработки алгоритма и написания программы, как в аудитории, так и самостоятельно в качестве домашних заданий. Примеры решения задач разбираются на лекциях и практических занятиях, при необходимости по наиболее трудным темам проводятся дополнительные консультации. Основная цель решения задач – сформировать алгоритмическое мышление. Для решения всех задач необходимо знать и понимать лекционный материал. Поэтому в процессе изучения дисциплины рекомендуется регулярное повторение пройденного лекционного материала. Материал, законспектированный на лекциях, необходимо дома еще раз прорабатывать и при необходимости дополнять информацией, полученной на консультациях, практических занятиях или из учебной литературы.

Большое внимание должно быть уделено выполнению домашней работы. В качестве заданий для самостоятельной работы дома студентам предлагаются задачи, аналогичные разобранным на лекциях и практических занятиях или немного более сложные, которые являются результатом объединения нескольких базовых задач. Задачи сдаются через систему Яндекс-контекст, обязательны к решению три задачи из каждого тура.

Для более глубокого усвоения материала в течение обучения проводятся консультации (при необходимости) по разбору заданий для самостоятельной работы, которые вызвали затруднения.

В конце каждого семестра студенты сдают зачет. Зачет выставляется автоматически при условии сдачи всех лабораторных работ, при необходимости может быть проведен устный зачет по вопросам из списка.