

МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение информационных технологий»

ПРОГРАММИРОВАНИЕ И ОСНОВЫ АЛГОРИТМИЗАЦИИ

*Методические рекомендации к практическим занятиям
для студентов направления подготовки
15.03.06 «Мехатроника и робототехника»
дневной формы обучения*



Могилев 2023

УДК 621.01
ББК 36.4
П87

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Программное обеспечение информационных технологий» «24» марта 2023 г., протокол № 8

Составитель канд. техн. наук, доц. Н. Н. Горбатенко

Рецензент канд. техн. наук, доц. Е. В. Ильюшина

Методические рекомендации разработаны на основе рабочей программы по дисциплине «Программирование и основы алгоритмизации» для студентов направления подготовки 15.03.06 «Мехатроника и робототехника» дневной формы обучения и предназначены для использования при проведении практических занятий.

Учебное издание

ПРОГРАММИРОВАНИЕ И ОСНОВЫ АЛГОРИТМИЗАЦИИ

Ответственный за выпуск

В. В. Кутузов

Корректор

И. В. Голубцова

Компьютерная верстка

Е. В. Ковалевская

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 21 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».

Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.

Пр-т Мира, 43, 212022, г. Могилев.

© Белорусско-Российский
университет, 2023

Содержание

1 Практическое занятие № 1. Линейный вычислительный процесс.....	4
2 Практическое занятие № 2. Ветвящийся вычислительный процесс.....	4
3 Практическое занятие № 3. Циклический вычислительный процесс.....	5
4 Практическое занятие № 4. Вспомогательные алгоритмы.....	6
5 Практическое занятие № 5. Алгоритмы работы со структурированными типами данных.....	7
Список литературы.....	8

1 Практическое занятие № 1. Линейный вычислительный процесс

Цель занятия: получение практических навыков разработки линейных вычислительных процессов.

Теоретические сведения и содержание работы приведены в [1, с. 31–37].
Индивидуальные задания [1, с. 35–37].

Контрольные вопросы

- 1 Дайте определение линейного вычислительного процесса.
- 2 Назовите основные этапы разработки программы, реализующей линейный вычислительный процесс.
- 3 Нарисуйте блок-схему линейного вычислительного процесса.
- 4 Какой тип имеет результат выражения $10/2$?
- 5 Какие ограничения существуют при возведении числа в произвольную степень?
- 6 Почему выражение $\frac{1}{2a}$ в программе должно быть записано как $1/(2a)$?
- 7 Почему для ввода данных требуется два объекта: *cout* и *cin*?
- 8 Можно ли в примере 2 изменить порядок вычислений (например, *a* вычислить раньше *b*)?

2 Практическое занятие № 2. Ветвящийся вычислительный процесс

Цель занятия: получение практических навыков разработки ветвящихся вычислительных процессов.

Теоретические сведения и содержание работы приведены в [1, с. 38–56].
Индивидуальные задания: задание 1 [1, с. 47–49], задание 2 [1, с. 55–56].

Контрольные вопросы

- 1 Дайте определение ветвящегося вычислительного процесса.
- 2 Назовите основные этапы разработки программы, реализующей ветвящийся вычислительный процесс.
- 3 Нарисуйте блок-схему линейного вычислительного процесса.
- 4 В какой ситуации в условном операторе используются скобки $\{ \}$?
- 5 Почему при записи $(x < z < y)$ не будет сообщения об ошибке?
- 6 Расставьте в порядке понижения приоритета следующие операции: +, &&, >=, ||, *, <, %.
- 7 Почему в примере 3 нет проверки $x > 1$?
- 8 Изобразите блок-схему примера 5 без использования сложного условия.

9 По какой ветке будет выполняться алгоритм примера 2, если x и y равны?

10 Какой тип должно иметь выражение в операторе *switch*?

11 Если переменная z в условии задачи может принимать четыре значения, в зависимости от того, попадает ли x в интервалы меньше 0, от 0 до корня квадратного от 2, от корня квадратного от 2 до 10, больше 10, может ли эта задача быть решена с помощью оператора *switch*?

12 Что будет, если пропустить оператор *break* во всех ветках, в первой ветке, в последней ветке?

13 Чему будет равно S после выполнения следующего оператора при $k = 0$:

$S = 10$;

switch k {

case 1: $S = 0$; *break*;

case 2: $S = 1$; *break*;

case 3: $S = 3$; *break*;

}

14 Определите, чему будет равно S после выполнения следующего оператора:

$S = 1$;

switch k {

case 1: $S = 10$; *break*;

case 0: $S = S + 1$; *break*;

case 2: $S = S + 3$; *break*;

default $S = 0$; *break*;

}

Если $k = 1$;

15 Почему в примере 8 отсутствует ветка *default*?

3 Практическое занятие № 3. Циклический вычислительный процесс

Цель занятия: получение практических навыков разработки циклических вычислительных процессов.

Теоретические сведения и содержание работы приведены в [1, с. 57–82].

Индивидуальные задания: задание 1 [1, с. 69], задание 2 [1, с. 76–77], задание 3 [1, с. 82].

Контрольные вопросы

1 Дайте определение циклического вычислительного процесса.

2 Назовите основные этапы разработки программы, реализующей циклический вычислительный процесс.

3 Нарисуйте блок-схемы циклических вычислительных процессов.

4 Как можно выйти досрочно из цикла?

5 Если начальное значение счетчика окажется меньше конечного значения, будет ли выполняться тело цикла хотя бы один раз?

6 Можно ли при поиске максимального значения в произвольной последовательности чисел первоначальное значения максимума задавать равное нулю и почему?

7 Как будет выглядеть блок-схема примера 3, если надо найти не минимум, а максимум?

8 Почему в примере 12 $m = n / 2$ – наибольший возможный делитель?

9 Объясните проверку условия в примере 12.

10 Объясните в примере 11 две последние операции в цикле.

11 Когда надо использовать цикл «до», а когда цикл «пока»?

12 К какому виду цикла относится цикл со счетчиком?

13 Какой из трех циклов является универсальным?

14 Укажите, в каких ситуациях вместо цикла со счетчиком приходится использовать цикл *while*.

15 Всегда ли цикл *while* можно заменить циклом *do*?

16 Какой оператор надо использовать, если надо досрочно выйти из цикла?

17 Что означает условие выхода из цикла в примере 15?

18 Что такое наибольший общий делитель и каком алгоритм его нахождения?

19 Как можно сократить пример 20?

20 Почему при вычислении суммы бесконечной последовательности можно ограничить количество членов?

21 Почему в проверке на достижение точности член ряда указан по модулю?

22 В какой ситуации при вычислении с заданной точностью, несмотря на увеличение члена ряда с ростом n , можно завершить вычисления в цикле?

23 Почему рекомендуется получать рекуррентное соотношение для вычисления суммы ряда, который содержит факториал и возведение в целую степень?

24 Можно ли для примера 23 построить рекуррентное соотношение?

25 Если в примере 25 условие выхода из цикла будет не значение элемента меньше ε , а значение разности меньше ε , количество слагаемых будет больше или меньше?

4 Практическое занятие № 4. Вспомогательные алгоритмы

Цель занятия: получение практических навыков разработки вспомогательных вычислительных алгоритмов.

Теоретические сведения и содержание работы приведены в [1, с. 83–109].

Индивидуальные задания: задание 1 [1, с. 93–94], задание 2 [1, с. 100–101], задание 3 [1, с. 108–109].

Контрольные вопросы

1 Дайте определение подпрограммы.

2 Какие виды подпрограмм существуют в языке C++?

3 Может ли функция не иметь параметров?

4 Могут ли в качестве фактических параметров использоваться выражения?

5 Какие правила соответствия должны выполняться для формальных и фактических параметров?

6 Что будет, если имя формального параметра совпадает с глобальным параметром?

7 В примере 3 в основной программе используется цикл с параметром i и в подпрограмме используется цикл с параметром i . Не запутается ли программа с такими совпадениями?

8 Можно ли в теле функции использовать несколько операторов *return*?

9 Можно ли из одной функции вызывать другую функцию?

10 Можно ли в примере 6 при вычислении *NOD* параметры A и B описать как параметры, передаваемые по ссылке?

11 Можно ли в примере 6 поменять местами описание *void*-функции *NOD* и *SOKR*?

12 Перечислите преимущества *void*-функций перед функциями.

13 Чем параметры, передаваемые по ссылке, отличаются от параметров, передаваемых по значению?

14 Можно ли в примере 6 использовать не *void*-функцию, а функцию?

15 В каких случаях *void*-функцию можно заменить на функцию?

16 Можно ли в качестве фактических параметров по ссылке использовать выражения?

17 В каких ситуациях удобнее использовать функцию вместо *void*-функции?

18 Какая подпрограмма называется рекурсивной?

19 Почему в рекурсивной подпрограмме отсутствуют операторы цикла?

20 Может ли в рекурсивной подпрограмме отсутствовать развилка?

21 Почему в примере 10 отсутствует операция $k = k + 1$?

22 Как будет работать рекурсивная подпрограмма в примере 10 при $eps = 0,1$ и $x = 1$?

23 Как будет работать рекурсивная подпрограмма в примере 11 при $k = 333$?

24 В примере 12 укажите параметры по ссылке и параметры по значению в процедуре *sumrec*.

5 Практическое занятие № 5. Алгоритмы работы со структурированными типами данных

Цель занятия: получение практических навыков работы со структурированными типами данных: массивами, строками, структурами.

Теоретические сведения и содержание работы приведены в [1, с. 110–146].

Индивидуальные задания: задание 1 [1, с. 125–126], задание 2 [1, с. 132–133], задание 3 [1, с. 145–146].

Контрольные вопросы

- 1 Если новый массив получен из элементов исходного, сколько элементов надо зарезервировать для нового массива?
- 2 В каких ситуациях при получении нового массива не надо использовать новый индекс?
- 3 Для чего в примере 13 используются переменные p и r ?
- 4 Если новый массив получен из двух исходных массивов, сколько элементов надо зарезервировать для нового массива?
- 5 Сколько элементов в массиве размером 4 на 5?
- 6 Какое условие выполняется для элементов ниже главной диагонали?
- 7 Укажите параметры по ссылке и параметры по значению в *void*-функции *obmen* из примера 15.
- 8 Можно ли в примере 16 обойтись без признака нулевого элемента k ?
- 9 Чем C-строки отличаются от обычного массива символов?
- 10 В чем заключается недостаток использования оператора ввода `>>` для строк?
- 11 Почему не рекомендуется смешивать для ввода оператор `>>` и *getline*? Как избежать ошибок при таком смешивании?
- 12 Какие преимущества имеет класс *string* перед C-строками?
- 13 Какую библиотеку надо подключать, чтобы использовать функции для объединения C-строк?
- 14 В какой библиотеке описан класс *string*?
- 15 Какие функции надо использовать, чтобы преобразовать строку в числовой тип?

Список литературы

- 1 **Ширёва, С. Н.** Основы программирования на языке C/C++: практикум / С. Н. Ширёва. – Екатеринбург: РГППУ, 2020. – 147 с.
- 2 **Павловская, Т. А.** C/C++. Программирование на языке высокого уровня: учебник для вузов / Т. А. Павловская. – Санкт-Петербург: Питер, 2020. – 432 с.
- 3 **Навроцкий, А. А.** Основы алгоритмизации и программирования в среде Visual C++: учебно-методическое пособие / А. А. Навроцкий. – Минск: БГУИР, 2021. – 160 с.
- 4 **Пацей, Н. В.** Основы алгоритмизации и программирования: учебно-методическое пособие для студентов специальности «Информационные системы и технологии (издательско-полиграфический комплекс)» / Н. В. Пацей. – Минск : БГТУ, 2022. – 289 с.