МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

**«БРАТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**Кафедра математики и физики**

УТВЕРЖДАЮ:
Проректор по учебной работе
_____Е.И.Луковникова
«_____» декабря 2018 г.

# РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

# ТЕОРИЯ АЛГОРИТМОВ

**Б1.В.09**

## НАПРАВЛЕНИЕ ПОДГОТОВКИ

**01.03.02 Прикладная математика и информатика**

## ПРОФИЛЬ ПОДГОТОВКИ
**Инженерия программного обеспечения**

Программа академического бакалавриата

Квалификация (степень) выпускника: бакалавр

**СОДЕРЖАНИЕ ПРОГРАММЫ** **Стр.**

## 1. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ, СООТНЕСЕННЫХ С ПЛАНИРУЕМЫМИ РЕЗУЛЬТАТАМИ ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

**Вид деятельности выпускника**

Дисциплина охватывает круг вопросов, относящихся к проектному и производственно-технологическому видам профессиональной деятельности выпускника в соответствии с компетенциями и видами деятельности, указанными в учебном плане..

**Цель дисциплины**

Целью изучения дисциплины является: ознакомление обучающихся с математическими моделями алгоритмов, основами теории сложности вычислений, привить практические навыки конструирования алгоритмов решения распространенных математических задач.

**Задачи дисциплины**

- формирование математической культуры обучающегося;
- обучение рациональным способам решения задач;
- формирование и развитие умений и навыков, позволяющих применять современные математические методы и программное обеспечение для решения задач науки и техники.

| Код компетенции | Содержание компетенций | Перечень планируемых результатов обучения по дисциплине |
|---|---|---|
| 1 | 2 | 3 |
| ОПК-1 | Способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой | **знать:**<br>- теоретические основы теории алгоритмов, базовые математические модели алгоритма, основы теории сложности;<br>**уметь:**<br>–использовать математические методы для анализа алгоритмов;<br>**владеть:**<br>- навыками применения аппарата математики и информатики для решения прикладных задач |
| ОПК-3 | Способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям | **знать:**<br>- основные алгоритмы обработки данных, основы программирования;<br>**уметь:**<br>–разрабатывать алгоритмические и программные решения для решения прикладных задач;<br>**владеть:**<br>– методами описания алгоритмов |

| | | |
|---|---|---|
| ПК-7 | Способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения | **знать:**<br>– средства разработки программных решений;<br>**уметь:**<br>– решать задачи теоретического и прикладного характера из различных разделов теории алгоритмов;<br>**владеть:**<br>– навыками разработки прикладных программ |

## 2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Дисциплина Б1.В.09 Теория алгоритмов относится к вариативной части.

Дисциплина Теория алгоритмов базируется на знаниях, полученных при изучении основных общеобразовательных программ.

Основываясь на изучении перечисленных дисциплин, Теория алгоритмов представляет основу для изучения дисциплин Численные методы, Методы оптимизации.

Такое системное междисциплинарное изучение направлено на достижение требуемого ФГОС уровня подготовки по квалификации бакалавр.

## 3. РАСПРЕДЕЛЕНИЕ ОБЪЕМА ДИСЦИПЛИНЫ

### 3.1. Распределение объема дисциплины по формам обучения

| *Форма обучения* | *Курс* | *Семестр* | *Трудоемкость дисциплины в часах* | | | | | | *Контрольная работа* | *Вид промежуточной аттестации* |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | *Всего часов (с экз.)* | *Аудиторных часов* | *Лекции* | *Лабораторные работы* | *Практические занятия* | *Самостоятельная работа* | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| **Очная** | 1 | 2 | 144 | 54 | 18 | 18 | 18 | 54 | - | Экзамен |
| **Заочная** | - | - | - | - | - | - | - | - | - | - |
| **Заочная** (ускоренное обучение) | - | - | - | - | - | - | - | - | - | - |
| **Очно-заочная** | - | - | - | - | - | - | - | - | - | - |

### 3.2. Распределение объема дисциплины по видам учебных занятий и трудоемкости

| *Вид учебных занятий* | *Трудо-емкость (час.)* | *в т.ч. в интерактивной, активной, иннова-ционной формах, (час.)* | *Распределение по семестрам, час* |
|---|---|---|---|
| | | | *2* |
| 1 | 2 | 3 | 4 |
| **I. Контактная работа обучающихся с преподавателем (всего)** | 54 | 24 | 36 |
| Лекции (Лк) | 18 | 6 | 18 |
| Лабораторные работы(ЛР) | 18 | - | 18 |
| Практические занятия (ПЗ) | 18 | 18 | 18 |
| Групповые (индивидуальные) консультации* | + | - | + |
| **II. Самостоятельная работа обучающихся (СР)** | 54 | - | 54 |
| Подготовка к лабораторным работам | 12 | - | 12 |
| Подготовка к практическим занятиям | 18 | - | 18 |
| Подготовка к экзамену в течение семестра | 24 | - | 24 |
| **III. Промежуточная аттестация** экзамен | 36 | - | 36 |
| Общая трудоемкость дисциплины час. | 144 | - | 144 |
| зач. ед. | 4,0 | - | 4,0 |

## 4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

### 4.1. Распределение разделов дисциплины по видам учебных занятий

**- для очной формы обучения:**

| *№ раз-дела и темы* | *Наименование раздела и тема дисциплины* | *Трудоем-кость, (час.)* | *Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость; (час.)* | | | |
|---|---|---|---|---|---|---|
| | | | *учебные занятия* | | | *самостоятельная работа обучающихся* |
| | | | *лекции* | *лабораторные работы* | *практические занятия* | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| **1.** | **Математические модели алгоритмов** | **64** | **10** | **10** | **10** | **34** |
| 1.1. | Основные понятия теории алгоритмов | 16 | 2 | 4 | 2 | 8 |
| 1.2. | Элементы формальной грамматики | 12 | 2 | - | 2 | 8 |
| 1.3. | Математические модели алгоритма | 26 | 4 | 6 | 6 | 10 |
| 1.4. | Алгоритмические проблемы | 10 | 2 | - | - | 8 |

| | | | | | | |
|---|---|---|---|---|---|---|
| **2.** | **Анализ эффективности алгоритма** | **44** | **8** | **8** | **8** | **20** |
| 2.1 | Введение в теорию сложности | 22 | 4 | 4 | 4 | 10 |
| 2.2 | Приближенные алгоритмы и эвристики | 22 | 4 | 4 | 4 | 10 |
| | **ИТОГО** | **108** | **18** | **18** | **18** | **54** |

### 4.2. Содержание дисциплины, структурированное по разделам и темам

| ***№ раздела и темы*** | ***Наименование раздела и темы дисциплины*** | ***Содержание лекционных занятий*** | ***Вид занятия в интерактивной, активной, инновационной формах, (час.)*** |
|---|---|---|---|
| **1** | **2** | **3** | **4** |
| **1.** | **Математические модели алгоритмов** | | |
| 1.1. | Основные понятия теории алгоритмов | Предмет изучения теории алгоритмов. Алгоритм и его свойства. Массовая задача. Способы описания алгоритма. Примеры алгоритмов. | - |
| 1.2. | Элементы формальной грамматики | Регулярные языки. Грамматики. Конечные автоматы. | Проблемная лекция( 2 час) |
| 1.3. | Модели алгоритма | Машина Тьюринга. Рекурсивные функции. Рекурсивные множества. Марковские подстановки. Нормально вычислимые функции и принцип нормализации Маркова | Лекция-беседа (2 часа) |
| 1.4. | Алгоритмические проблемы | Мощность множества. Сравнение мощностей. Нумерация машин Тьюринга. Мощность множества целочисленных функций. Проблема распознавания применимости и самоприменимости. Задача эквивалентности двух слов в ассоциативном исчислении. Проблема распознавания выводимости. | - |
| **2.** | **Анализ эффективности алгоритма** | | |
| 2.1 | Введение в теорию сложности | Виды эффективности алгоритма. Экспериментальная оценка сложности. Теоретические оценки сложности алгоритма. Теоремы о соотношении пространственной и временной эффективности. Асимптотическая сложность. Классы асимптотической сложности. | - |

| | | | |
|---|---|---|---|
| 2.2 | Приближенные алгоритмы и эвристики | Полиномиальная сводимость. Задача оптимизации. Приближенные алгоритмы. Мультипликативная ошибка. Эвристики. | Лекция-беседа (2 часа) |

### 4.3. Лабораторные работы

| *№ п/п* | *Номер раздела дисциплины* | *Наименование лабораторной работы* | *Объем (час.)* | *Вид занятия в интерактивной, активной, инновационной формах, (час.)* |
|---|---|---|---|---|
| 1 | **1.** | Конструирование алгоритмов | 10 | - |
| 2. | **2.** | Оценки эффективности алгоритмов | 8 | - |
| | | **ИТОГО** | **18** | **-** |

### 4.4 Практические занятия

| *№ п/п* | *Номер раздела дисциплины* | *Наименование тем семинаров практических занятий* | *Объем (час.)* | *Вид занятия в интерактивной, активной, инновационной формах, (час.)* |
|---|---|---|---|---|
| 1 | **1.** | Формальные модели алгоритма | 10 | Тренинг в малой группе (10 час ) |
| 2 | **2.** | Классы сложности алгоритма | 8 | Тренинг в малой группе (8 час ) |
| | | **ИТОГО** | **18** | **18** |

### 4.5. Контрольные мероприятия

Учебным планом не предусмотрено

## 5. МАТРИЦА СООТНЕСЕНИЯ РАЗДЕЛОВ УЧЕБНОЙ ДИСЦИПЛИНЫ К ФОРМИРУЕМЫМ В НИХ КОМПЕТЕНЦИЯМ И ОЦЕНКЕ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ

| *Компетенции* / *№, наименование разделов дисциплины* | *Кол-во часов* | *Компетенции* | | | *Σ комп.* | $t_{ср}$, *час* | *Вид учебных занятий* | *Оценка результатов* |
|---|---|---|---|---|---|---|---|---|
| | | *ОПК* | | *ПК* | | | | |
| | | *1* | *3* | *7* | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| **1.** Математические модели алгоритмов | 64 | + | + | - | 2 | 32 | Лк,ЛР, ПЗ | Экзамен |
| **2.** Анализ эффективности алгоритма | 44 | - | - | + | 1 | 44 | Лк,ЛР, ПЗ | Экзамен |
| ***всего часов*** | **108** | **32** | **32** | **44** | **3** | **36** | | |

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

1. Ратинская, Е. В. Теория алгоритмов : учебное пособие / Е. В. Ратинская. - Братск : БрГУ, 2011. - 83 с. Рекомендации для самостоятельной работы – стр. 21-55.

## 7. ПЕРЕЧЕНЬ ОСНОВНОЙ И ДОПОЛНИТЕЛЬНОЙ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

| № | *Наименование издания* | *Вид занятия* | *Количество экземпляров в библиотеке*, шт. | *Обеспеченность, (экз./ чел.)* |
|---|---|---|---|---|
| **1** | **2** | **3** | **4** | **5** |
| | **Основная литература** | | | |
| 1. | Информатика. Базовый курс : учебник для бакалавров и специалистов / Под ред. С. В. Симоновича. - 3-е изд. - Санкт-Петербург : Питер, 2015. - 640 с. - (Учебник для вузов. Стандарт третьего поколения) . | Лк, ПЗ | 123 включая аналоги | 1,0 |
| 2 | Незнанов, А. А. Программирование и алгоритмизация : учебник / А. А. Незнанов. - Москва : Академия, 2010. - 304 с. | ЛР, СР | 10 | 0,5 |
| 3. | Ратинская, Е. В. Теория алгоритмов : учебное пособие / Е. В. Ратинская. - Братск : БрГУ, 2011. - 83 с. | Лк, ЛР, ПЗ, СР | 69 | 1,0 |
| | **Дополнительная литература** | | | |
| 4. | Глухов, М. М. Математическая логика. Дискретные функции. Теория алгоритмов : учебное пособие / М. М. Глухов, А. Б. Шишков. - Санкт-Петербург : Лань, 2012. - 416 с | СР | 6 | 0,3 |

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

1.Электронный каталог библиотеки БрГУ
http://irbis.brstu.ru/CGI/irbis64r_15/cgiirbis_64.exe?LNG=&C21COM=F&I21DBN=BOOK&P21DBN=BOOK&S21CNR=&Z21ID=.
2. Электронная библиотека БрГУ
http://ecat.brstu.ru/catalog .
3. Электронно-библиотечная система «Университетская библиотека online»
http://biblioclub.ru .
4. Электронно-библиотечная система «Издательство «Лань»
http://e.lanbook.com .
5. Информационная система "Единое окно доступа к образовательным ресурсам"
http://window.edu.ru .
6. Научная электронная библиотека eLIBRARY.RU http://elibrary.ru .
7. Университетская информационная система РОССИЯ (УИС РОССИЯ)
https://uisrussia.msu.ru/ .
8. Национальная электронная библиотека НЭБ
http://xn--90ax2c.xn--p1ai/how-to-search /.
Специальные тематические сайты
Сеть разработчиков MicroSoft https://docs.microsoft.com/ru-ru/

## 9. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Обучающийся должен разработать собственный режим равномерного освоения дисциплины. Подготовка студента к предстоящей лекции включает в себя ряд важных познавательно-практических этапов:

– чтение записей, сделанных в процессе слушания и конспектирования предыдущей лекции, вынесение на поля всего, что требуется при дальнейшей работе с конспектом и учебником;

– техническое оформление записей (подчеркивание, выделение главного, выводов, доказательств);

– выполнение практических заданий преподавателя;

– знакомство с материалом предстоящей лекции по учебнику и дополнительной литературе.

Наиболее продуктивной является самостоятельная работа. Она складывается из чтения учебников и методических пособий, решения задач, выполнения контрольных заданий. Студент должен помнить, что только при систематической и упорной самостоятельной работе можно качественно освоить учебный материал.

Завершающим этапом изучения данной дисциплины в соответствии с учебным планом является сдача экзамена. На экзамене студент должен: проявить умение применять теоретические сведения к решению задач построение и анализ алгоритмов; знание теоретических основ курса на уровне определений, теорем, формул; умение выбирать методы анализа и оценки выбранных решений.

### 9.1. Методические указания для обучающихся по выполнению лабораторных и практических работ

**Практическое занятие №1**

Формальные модели алгоритма

Цель работы:

Приобрести навыки описания основных алгоритмов решения распространенных задач.

Задание 1:

Ниже записан алгоритм. Получив на вход число x, этот алгоритм печатает число L. Укажите наибольшее нечетное число x, при вводе которого алгоритм печатает 53.

```
var x, L, M, D: integer;
begin
  readln(x);
  D:=x;
  L:=23;
  M:=141;
  while L<=M do
  begin
    L:=L+D;
    M:=M-3*D;
  end;
  writeln(L);
end.
```

Решение:
Разберем решение по одному из доступных вариантов выполнения данного задания.
Рассмотрим алгоритм:
Результатом программы является вывод L.

Цикл перестанет работать, когда L станет больше M (т.к. while L<=M).

D в программе - это и есть искомый x (D:=x;)

В цикле строка L:=L+D работает, как сумматор: L накапливает в себе сумму всех D, которая в нашей задаче не меняется и равна введенному x

Так как сумматор (L) по правилам до цикла обычно обнуляется, то сделаем также: т.е. в строке 5 вместо L:=23 представим, что L:=0. Тогда и условие задачи поменяется: т.е. вместо указанного в условии числа 53 программа выводит L равное 53-23 = 30

Так как по заданию необходимо найти наибольший x, то представим, что он равен как раз 30:
L = L + D => L = 0 + 30 => L=30

После первого прохода по циклу условие продолжает работать: т.е. L<=M -> истинно (30<51), значит одного прохождения цикла недостаточно, и, кроме того, 30 - четное (по заданию x - нечетное)

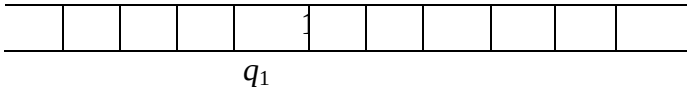
За два прохождения цикла L увеличится на 2*D, то есть нужно взять такое D, чтобы 2*D было равно 30. Значит 30/2 = 15. То есть D = x = 15

Результат: 15

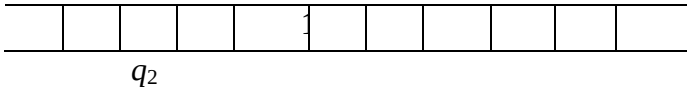
Задание 2. Изучить работы машины Тьюринга, заданной функциональной схемой

| | $a_0$ | 1 | + |
|---|---|---|---|
| $q_1$ | $a_0q_0$ | $a_0\ q_2$Л | $a_0\ q_2$Л |
| $q_2$ | $1q_0$ | $1q_2$Л | $1q_3$П |
| $q_3$ | $a_0q_1$Л | $1q_3$П | - |

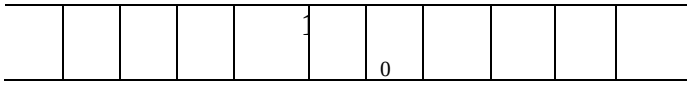
Решение
Пусть на ленте записана начальная информация "11+1" (рис 3.2.):




На первом такте действует команда $1q_1 \rightarrow a_0q_2$Л. В результате её выполнения, машина Тьюринга стирает последнюю единицу, перемещается на шаг влево и переходит в состояние $q_2$:




На втором такте действует команда $+q_2 \rightarrow 1q_3$П, которая заменяет знак «+» на «1» и сдвигает УУ на ячейку вправо. Так так эта ячейка пуста, помечаем её символом $a_0$. На ленте создается конфигурация:




На третьем такте выполняется команда $a_0q_3 \rightarrow a_0q_1$Л (рис. 3.5.):

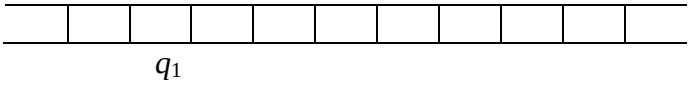
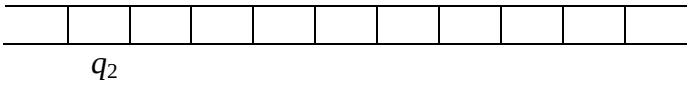



*Рис. 3.5. Третий такт работы машины Тьюринга*

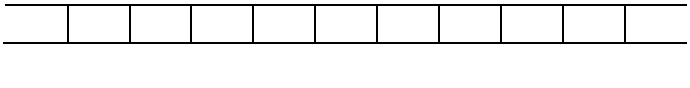
На 4-м этапе действует та же команда, что и в начале: $1q_1 \rightarrow a_0q_2$Л.. Результат её работы мы можем наблюдать на рис. 3.6.




На 5-м и 6-м шагах положение УУ постепенно смещается к левому краю, не меняя при этом информации, записанной на ленте. После шестого шага конфигурация машины Тьюринга принимает вид:


*Рис 3.7. Состояние машины Тьюринга после шестого такта*

Здесь выполняется последняя команда $a_0q_2 \rightarrow 1q_0$, после которой машина переходит в стоп-состояние:




В результате работы программы слово «11+1» было преобразовано в слово «111».

***Пример 2.*** Программа машины Тьюринга, реализующая следующий алгоритм: из *n* записанных на ленте единиц машина оставляет (*n*-2) единицы, если $n \geq 2$ и работает вечно, если *n*=0 или *n*=1, имеет вид:

*Таблица 3.2*

| | $a_0$ | 1 |
|---|---|---|
| $q_1$ | $a_0q_1$П | $a_0q_2$Л |
| $q_2$ | $a_0q_2$П | $a_0q_3$Л |
| $q_3$ | $a_0q_0H$ | $1q_0$Н |

Посмотрим, как она переработает слово «111» (начальное состояние стандартное). Текущую конфигурацию машины Тьюринга обычно записывают в строчку, помещая обозначение состояния справа от обозреваемого символа. Получим следующую последовательность выполнения шагов:

1) $111q_1$;
2) $11q_2$;
3) $1q_3$;
4) $1\ q_0$.

Форма отчетности:

Выполнить задание в тетради и использовать его при подготовке к экзамену и контрольной работы

Задания для самостоятельной работы:

1. Ниже записан алгоритм. Получив на вход число x, этот алгоритм печатает два числа: L и M. *Укажите наименьшее число x, при вводе которого алгоритм печатает сначала 5, а потом 7.*

```
var x, L, M: integer;
begin
readln(x);
L := 0;
M := 0;
while x>0 do
```

```
begin
M := M + 1;
if x mod 2 <> 0 then
L := L + 1;
x := x div 2;
end;
writeln(L);
writeln(M);
end.
```

1. Составьте алгоритм деления отрезка на *k* равных частей.

2. Опишите алгоритм построения касательной к окружности, проведенной через заданную точку, лежащую вне этой окружности.

3. Предложите алгоритм поиска элемента:
а) в неупорядоченном массиве;
б) в полностью упорядоченном массиве без повторяющихся элементов.

4. Опишите алгоритм Евклида для нахождения наибольшего общего делителя.

5. Опишите алгоритм, определяющий для произвольного натурального числа *N*, является ли оно составным.

6. Напишите программу для машины Тьюринга, которая к *n* записанным на ленте единицам
а) прибавляла бы ещё одну,
б) прибавляла бы ещё 2.

7. Проверьте, что программа машины Тьюринга (начальное состояние – стандартное), реализующая алгоритм вычисления функции $\varphi(n) = 2n$, имеет вид

| | $q_1$ | $q_2$ | $q_3$ | $q_4$ |
|---|---|---|---|---|
| $a_0$=0 | $a\ q_2$ Л | 0 $q_0$Н | 0 $q_4$ П | 0 $q_0$Н |
| 1 | 1$q_1$ П | $b\ q_3$ Л | 1 $q_1$ Н | 1 $q_0$ Н |
| $a$ | $a\ q_1$ П | $a\ q_2$ Л | $a\ q_0$ Н | 1 $q_4$П |
| $b$ | $b\ q_1$ П | $b\ q_2$ Л | $a\ q_0$ Н | 1 $q_4$П |

8. Выясните, в какое слово переработает слово $A$ машина Тьюринга, заданная функциональной схемой:
a) $A$=1010111;

| | $a_0$ | 0 | \| |
|---|---|---|---|
| $q_1$ | $a_0q_2$Л | \|$q_2$Л | \| $q_1$Л |
| $q_2$ | $a_0q_0$Н | $a_0q_3$Л | 0$q_1$Л |
| $q_3$ | $a_0q_0$Н | 0$q_1$Л | \| $q_3$Л |

b) $A$=11111;

| | $a_0$ | \| |
|---|---|---|
| $q_1$ | \|$q_2$Л | \|$q_1$Л |
| $q_2$ | \|$q_3$Л | \|$q_2$Л |
| $q_3$ | 1$q_0$Н | \|$q_3$Л |

c) $A$=11

| | $a_0$ | \| | α | β |
|---|---|---|---|---|
| $q_1$ | α $q_2$Л | \| $q_1$П | α $q_1$П | β$q_1$П |
| $q_2$ | | β $q_3$Л | α $q_2$Л | β$q_2$Л |
| $q_3$ | $a_0q_4$П | \| $q_1$Н | | |
| $q_4$ | $a_0q_0$Н | | \| $q_4$П | \| $q_4$П |

d) $A$=11;

| | $a_0$ | \| | α | β |
|---|---|---|---|---|
| $q_1$ | $a_0q_4$П | α $q_2$П | | |
| $q_2$ | β$q_3$Л | | α $q_2$П | β$q_2$П |
| $q_3$ | $a_0q_4$П | \|$q_1$Н | α $q_3$Л | β$q_3$Л |
| $q_4$ | \| $q_5$П | | \| $q_4$П | \| $q_4$П |
| $q_5$ | \| $q_0$Н | | | |

e) $A$=11;

| | $a_0$ | \| | α | β |
|---|---|---|---|---|
| $q_1$ | $a_0q_4$П | β$q_2$Л | α $q_1$Л | |
| $q_2$ | α $q_3$П | \|$q_2$Л | α $q_2$Л | |
| $q_3$ | | \| $q_3$П | α $q_3$П | β$q_1$Л |
| $q_4$ | \| $q_0$Н | | \| $q_4$П | \| $q_4$П |

9. Определите, какую функцию вычисляют машины Тьюринга, заданные функциональной схемой:

a)

| | $a_0$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $q_1$ | 1$q_0H$ | 1$q_0H$ | 2$q_0H$ | 3$q_0H$ | 4$q_0H$ | 5$q_0H$ | 6$q_0H$ | 7$q_0H$ | 8$q_0H$ | 9$q_0H$ | 0$q_1$Л |

b)

| | $a_0$ | 1 |
|---|---|---|
| $q_1$ | 1$q_2$П | 1$q_1$Л |
| $q_2$ | 1$q_0$ | 1$q_0$ |

c)

| | $a_0$ | 0 | 1 |
|---|---|---|---|
| $q_1$ | $a_0q_0$ | 0$q_2$Л | 1$q_1$Л |
| $q_2$ | $a_0q_0$ | 0$q_2$Л | 0$q_3$П |
| $q_3$ | | 1$q_4$П | |
| $q_4$ | $a_0$ $q_1$Л | 0$q_4$П | 1$q_4$П |

d)

| | $a_0$ | \| | α | β |
|---|---|---|---|---|
| $q_1$ | $a_0q_4$П | α $q_2$П | | |
| $q_2$ | β$q_3$Л | | α $q_2$П | β$q_2$П |
| $q_3$ | $a_0q_4$П | \| $q_1$Н | α $q_3$Л | β$q_4$Л |
| $q_4$ | \| $q_5$П | | \| $q_4$П | \| $q_4$П |
| $q_5$ | \| $q_0$П | | | |

10. Составьте программу машины Тьюринга, реализующую алгоритм вычисления функции $sgn(x) = \begin{cases} 0, & \text{если } x = 0 \\ 1, & \text{если } x \neq 0 \end{cases}$.

11. Напишите программу для машины Тьюринга, позволяющую выполнить следующие преобразования слов в алфавите $A$={0, 1}:
   a) заменить во входном слове в алфавите все буквы 0 на 1 и наоборот;
   b) переместить 0 через блок из единиц (011..11 →11..110);
   c) во входном слове из 0 и 1 переместить первую букву в конец слова;

d) удвоить слово, состоящее из единиц (используйте дополнительную букву в качестве аналога единицы);

e) записать слово из 0 и 1 в обратном порядке;

f) прибавить 1 к натуральному числу, записанному в двоичной записи;

12. Пусть аргументы на ленте записаны в унарной записи: 11=2, 111=3, 1111=4 и т. д. Напишите программы машин Тьюринга, вычисляющих следующие функции натурального аргумента:

a. $f(x) = \begin{cases} x-1, & x > 0, \\ 0, & x = 0; \end{cases}$

b. $f(x) = \begin{cases} 0, & \text{если } x \text{ четное}, \\ 1, & \text{если } x \text{ нечетное}; \end{cases}$

c. $f(x, y) = | x - y |$ (числа $x$ и $y$ на ленте разделены одним пробелом);

d. $f(x, y) = max(x, y)$;

e. $f(x) \equiv x \, mod 3$;

f. $f(x) = [x/2]$

13. Нормальный алгоритм в алфавите $A$={0,1,2,3,4,5,6,7,8,9,*} задается схемой:

Λ*→ ·1, 9Λ→ *0,
0*→ ·1, 0Λ→ ·1,
1*→ ·2, 1Λ→ ·2,
2*→ ·3, 2Λ→ ·3,
3*→ ·4, 3Λ→ ·4,
4*→ ·5, 4Λ→ ·5,
5*→ ·6, 5Λ→ ·6,
6*→ ·7, 6Λ→ ·7,
7*→ ·8, 7Λ→ ·8,
8*→ ·9, 8Λ→ ·9.
9*→ *0,

В какое слово будет переработано слово: а) 145; б) 79; в) 999, г) 392?

14. Нормальный алгоритм в алфавите $A$={1,2,3} задается схемой: 11 → 22 , 12 → 23, 13 → 21. В какое слово будет переработано слово: а) 32211; б) 321; в) 332112, г) 3112?

15. Задан алфавит $A$={$a$,$b$,$c$}. Составьте схему нормального алгоритма, позволяющего:

1. в слове P заменить все пары *ab* на *c*;
2. в слове P заменить первую пару *ab* на *c*;
3. заменить слово *P* на пустое слово (удалить все символы из *P*);
4. заменить любое слово *P* на слово *a*;
5. выбрать символ, стоящий на *i*-м месте (получить слово, состоящее из одного символа);
6. поменять символы местами так, чтобы сначала шли все буквы *a*, потом *b* и затем все *c*;
7. переместить первый символ в конец слова;
8. перевернуть слово, записав его в обратном порядке;
9. удвоить каждый символ в слове *P* (например, *babc→bbaabbcc*).

Рекомендации по выполнению заданий и подготовке к практическому занятию

1. Ознакомиться с заданием;
2. Изучить теоретические сведения, полученные на лекции;
3. Ознакомиться с примерами решения подобных задач в учебной литературе;
4. Выполнить задание в тетради.

Основная литература

1. Информатика. Базовый курс : учебник для бакалавров и специалистов / Под ред. С. В. Симоновича. - 3-е изд. - Санкт-Петербург : Питер, 2015. - 640 с. - (Учебник для вузов. Стандарт третьего поколения).
2. Ратинская, Е. В. Теория алгоритмов : учебное пособие / Е. В. Ратинская. - Братск : БрГУ, 2011. - 83 с.

Контрольные вопросы для самопроверки

1. Какие способы описания алгоритмов Вы знаете?
2. Дайте определение целевой функции.
3. Что такое платежная матрица?

## Практическое занятие № 2

Классы сложности алгоритма

Цель работы:

Научиться определять асимптотический класс сложности алгоритма по его описанию.

Задание 1:

1. Оценить временную сложность алгоритма Прима поиска остова минимального веса в графе.

Напомним, что остовом в связном графе называется такой подграф, который содержит все вершины исходного графа, но не имеет циклов.

Пусть граф описан матрицей смежности.

**Решение.** Выберем любую вершину графа и включим её в остов. Рассмотрим связанные с ней ребра. Найдём вершину, смежную с той, что включена в остов ребром минимального веса, и тоже включим её в остов.

Для вершин, включённых в остов, рассмотрим связанные с ними рёбра, ведущие к вершинам, не включённым в остов. Выберем вершину, связанную ребром минимального веса и включим её в остов. Эту процедуру продолжим до тех пор, пока все вершины не будут включены в остов.

Для матричного представления алгоритм примет следующий вид.

Выберем любую вершину и в соответствующей ей строке матрицы найдём ячейку с минимальным весом. Эта процедура потребует $n$ сравнений, где $n$-порядок матрицы (число вершин в графе). Имя столбца этой ячейки определяет вершину, которую нужно включить в остов. Объединим эти вершины в одну, соответствующую вершинам, включённым в остов. Для этого рассмотрим строки и столбцы этих вершин. Если в $i$-й ячейке одной из строк не нулевое число, оно запишется в результирующую строку; если ненулевое значение в обеих строках, то в результирующую строку запишем минимальное. Таким образом, для «обработки» одной вершины потребуется число операций, кратных $2n$. Число вершин в рассматриваемом графе сократится на единицу. Повторим то же самое для результирующей строки, пока число вершин не станет равным единице.

Ответ: Общее количество операций оценивается числом, пропорциональным $n^2$, т.е. оценка сложности алгоритма имеет квадратичную сложность $\delta = C \cdot n^2$

Задание 2:

Оценить сложность алгоритма Прима поиска остова минимального веса в графе.

Напомним, что остовом в связном графе называется такой подграф, который содержит все вершины исходного графа, но не имеет циклов.

Пусть граф описан матрицей смежности.

**Решение.** Выберем любую вершину графа и включим её в остов. Рассмотрим связанные с ней ребра. Найдём вершину, смежную с той, что включена в остов ребром минимального веса, и тоже включим её в остов.

Для вершин, включённых в остов, рассмотрим связанные с ними рёбра, ведущие к вершинам, не включённым в остов. Выберем вершину, связанную ребром минимального веса и включим её в остов. Эту процедуру продолжим до тех пор, пока все вершины не будут включены в остов.

Для матричного представления алгоритм примет следующий вид.

Выберем любую вершину и в соответствующей ей строке матрицы найдём ячейку с минимальным весом. Эта процедура потребует $n$ сравнений, где $n$-порядок матрицы (число вершин в графе). Имя столбца этой ячейки определяет вершину, которую нужно включить в остов. Объединим эти вершины в одну, соответствующую вершинам, включённым в остов. Для этого рассмотрим строки и столбцы этих вершин. Если в $i$-й ячейке одной из строк не нулевое число, оно запишется в результирующую строку; если ненулевое значение в обеих строках, то в результирующую строку запишем минимальное. Таким образом, для «обработки» одной вершины потребуется число операций, кратных $2n$. Число вершин в рассматриваемом графе сократится на единицу. Повторим то же самое для результирующей строки, пока число вершин не станет равным единице.

Ответ: Общее количество операций оценивается числом, пропорциональным $n^2$, т.е. оценка сложности алгоритма имеет квадратичную сложность $\delta = C \cdot n^2$

Задания для самостоятельной работы:

1. Оцените эффективность различных алгоритмов сортировки: пузырьком, выбором максимального (минимального) элемента, простыми вставками.
2. Сравните эффективность алгоритмов вычисления определителя методами разложения и методом Гаусса.
3. Оцените сложность(О-нотацию) алгоритма:
   1) поиска максимального элемента в одномерном неотсортированном массиве;
   2) поиска заданного элемента в отсортированном массиве;
   3) циклического сдвига в массиве;
   4) алгоритма Евклида нахождения наибольшего общего делителя двух чисел.
4. Определите класс сложности алгоритмов решения следующих задач:
   1) Поиск общих элементов двух массивов.
   2) Подсчет различных букв в слове.
   3) Решение системы линейных уравнений.
   4) Нахождение остатка от деления числа $n$ на m.
   5) Решение алгебраического уравнения методом дихотомии.
   6) Нахождение кратчайшего пути в графе.
   7) Вычисление $k$-го числа Фибоначчи.
5. Сравните сложность алгоритмов вычисления тангенса угла с помощью разложения в ряд Маклорена и в цепную дробь ($\mathrm{tg}x = x/(1-x^2/(3-x^2/(5-x^2/(7-x^2/...))))$ ).
6. Упростите алгоритм для вычисления значения выражения $a_0+a_1x+a_2x^2+\ldots+a_nx^n$ при заданном $x$.
7. Подберите упрощенный алгоритм возведения натурального числа в $n$-ю степень.

8. Оцените эффективность различных алгоритмов сортировки: пузырьком, выбором максимального (минимального) элемента, простыми вставками.
9. Сравните эффективность алгоритмов вычисления определителя методами разложения и методом Гаусса.
10. Оцените сложность(О-нотацию) алгоритма:
    1) поиска максимального элемента в одномерном неотсортированном массиве;
    2) поиска заданного элемента в отсортированном массиве;
    3) циклического сдвига в массиве;
    4) алгоритма Евклида нахождения наибольшего общего делителя двух чисел.
11. Определите класс сложности алгоритмов решения следующих задач:
    1) Поиск общих элементов двух массивов.
    2) Подсчет различных букв в слове.
    3) Решение системы линейных уравнений.
    4) Нахождение остатка от деления числа *n* на m.
    5) Решение алгебраического уравнения методом дихотомии.
    6) Нахождение кратчайшего пути в графе.
    7) Вычисление *k*-го числа Фибоначчи.

Рекомендации по выполнению заданий и подготовке к практическому занятию

1. Ознакомиться с заданием;
2. Изучить теоретические сведения, полученные на лекции;
3. Ознакомиться с примерами решения подобных задач в учебной литературе;
4. Выполнить задание в тетради.

Основная литература

1. Информатика. Базовый курс : учебник для бакалавров и специалистов / Под ред. С. В. Симоновича. - 3-е изд. - Санкт-Петербург : Питер, 2015. - 640 с. - (Учебник для вузов. Стандарт третьего поколения).
2. Ратинская, Е. В. Теория алгоритмов : учебное пособие / Е. В. Ратинская. - Братск : БрГУ, 2011. - 83 с.

Контрольные вопросы для самопроверки

1. Что такое пространственная эффективность алгоритма?
2. Что такое временная эффективность алгоритма?
3. Какие классы сложности относят к практическим?
4. Что означает нотация $O(n^2)$?

**Лабораторная работа №1**

Конструирование алгоритмов

Цель работы: закрепление теоретического материала; развитие алгоритмического и логического мышления; совершенствование навыков программирования.

Задание**:**

Найти наибольший общий делитель двух чисел M и N

Порядок выполнения:

1. Для начала разберемся, что это и как это работает.

Алгоритм Евклида позволяет найти нам наибольший общий делитель чисел. Как это работает:

Пусть M = 18, N = 30.

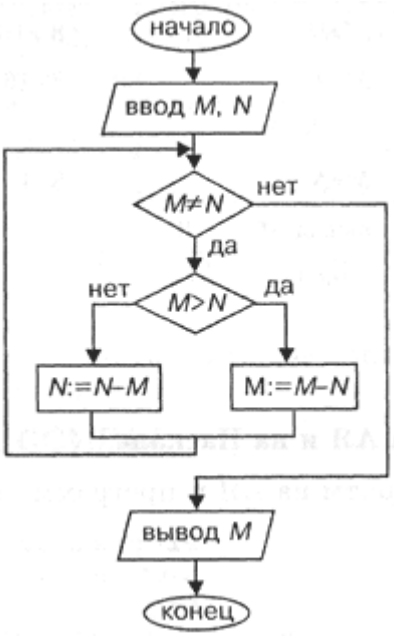
Цикл: M!=0 and N!=0

Если M > N то M = M %N, если меньше, то N = N % M, таким образом мы сначала находим остаток деления, а потом повторяем действия. У нас M < b, значит, ищем остаток деления N % M (30 % 18) = 12, присваиваем N = 12, повторяем цикл но теперь у нас уже M > N(N = 12)
значит выполняем M % N (18 % 12) = 6? снова переходим к циклу, теперь снова N > M, значит выполняем N % M (30 % 6), остаток от деления 0, на этом мы прекращаем наш цикл и узнаем, что наибольший общий делитель 18 и 30 = 6. и выводим M + N (N = 0, M = 6).

2.Описание алгоритма нахождения НОД делением

1. Большее число делим на меньшее.
2. Если делится без остатка, то меньшее число и есть НОД (следует выйти из цикла).
3. Если есть остаток, то большее число заменяем на остаток от деления.
4. Переходим к пункту 1.

3. Про описанию составим блок-схему алгоритма:




4. Напишем программу, реализующую алгоритм:

```
int gcd(int a, int b) {
  int c;
  while (b) {
     c = a % b;
     a = b;
     b = c;
   }
  return abs(a);
}
```

Форма отчетности:

Отчет по лабораторной работке должен содержать:

1. Наименование лабораторной работы;
2. Разработанную программу;
3. Результаты её тестирования;
4. Выводы по работе.

Задания для самостоятельной работы:

1. Пусть $\bar{a}$-вектор в $n$-мерном линейном пространстве, $M(n \times n)$ - матрица линейного оператора $f$. Нужно найти координаты вектора $f(\bar{a})$. Составьте программу, вычисляющую $f(\bar{a})$.

Входные данные: размерность пространства $n$ ($n$<11), координаты вектора $\bar{a}$, коэффициенты матрицы $M$.

Выходные данные: координаты вектора $f(\bar{a})$.

2. Алгоритм Евклида используется для нахождения наибольшего общего делителя двух натуральных чисел. Составьте программу, реализующую этот алгоритм.

Входные данные: натуральные числа $n$ и $m$ ($n$, $m \leq 1000$).

Выходные данные: НОД($n$, $m$).

3. Координаты вектора.

Пусть $\bar{a}$-вектор в $n$-мерном линейном пространстве, $M(n \times n)$ - матрица линейного оператора $f$. Нужно найти координаты вектора $f(\bar{a})$. Составьте программу, вычисляющую $f(\bar{a})$.

Входные данные: размерность пространства $n$ ($n$<11), координаты вектора $\bar{a}$, коэффициенты матрицы $M$.

Выходные данные: координаты вектора $f(\bar{a})$

4. Числами Фибоначчи называется последовательность $a_0, a_1, ..., a_n, ...$, где $a_0 = 0$, $a_1 = 1$, $a_k = a_{k-1} + a_{k-2}$ $(k > 1)$.

Начало ряда: 0,1,1,2,3,5,8…

Требуется найти $N$-е число Фибоначчи.

Входные данные: целое число $N$ ($0 \leq N \leq 30$).

Выходные данные : $N$-е число Фибоначчи.

Рекомендации по выполнению заданий и подготовке к лабораторной работе

1. Ознакомиться с заданием;
2. Изучить теоретические сведения, полученные на лекции;
3. Ознакомиться с примерами решения подобных задач в учебной литературе;
4. Выполнить задание.

Рекомендуемые источники

Основная литература

1. Незнанов, А. А. Программирование и алгоритмизация : учебник / А. А. Незнанов. - Москва : Академия, 2010. - 304 с.

2. Ратинская, Е. В. Теория алгоритмов : учебное пособие / Е. В. Ратинская. - Братск : БрГУ, 2011. - 83 с.

Контрольные вопросы для самопроверки

1.Что такое массовая задача?.

2. Какие Вы знаете способы описания алгоритмов?

3. Какими свойствами должен обладать алгоритм?

## **Лабораторная работа №2**

Оценки эффективности алгоритмов

Цель работы: Научиться оценивать эффективность реализации алгоритма.

Задание **:**

Требуется провести анализ и оценку временной сложности алгоритма пузырьковой сортировки:

Порядок выполнения:

1. Составьте алгоритм решения задачи.
2. Определите параметр задачи *n*, по порядку которой будет оцениваться сложность алгоритма.
3. Определите О-нотацию. Для этого выясните, как изменяется время работы или количество операций при линейном увеличении параметра *n*.
4. Составьте программу, реализующую данный алгоритм.
5. Протестируйте программу на нескольких примерах с разными значениями параметра *n*. Тестируемые параметры должны отличаться на порядок.
6. Укажите, к какому классу сложности относится алгоритм.
7. В отчете представить: описание алгоритма на бумажном или электронном носителе, расчет

   О-нотации, таблицу, показывающую зависимость между временем работы программы и значением параметра:

| Параметр | $n_1$ | $n_2$ | $n_3$ |
|---|---|---|---|
| Время исполнения | | | |

1. Алгоритмы сортировки оценивают по следующим критериям:

- Время сортировки – основной параметр, описывающий время исполнения алгоритма вычислительной системой. Обычно оценивается в терминах вычислительной сложности O(N), где N – размер обрабатываемого контейнера данных. Для алгоритмов сортировки важны оценки для лучшего, среднего и худшего случаев, описываемых с точки зрения предварительного состояния данных. Хорошими алгоритмами сортировки считаются алгоритмы со сложностью O(N·logN) в среднем случае. Теория алгоритмов доказывает, что алгоритм сортировки, использующий только операцию сравнения ключей, не может обладать сложностью меньшей, чем O(N·logN).
- Размер дополнительной памяти – размер дополнительной памяти, используемой алгоритмом для размещения дополнительных данных, требуемых для выполнения сортировки, выраженный в байтах. В этот размер не включается занимаемое исходным контейнером пространство памяти.
- Устойчивость (*stability*) – алгоритм сортировки называется устойчивым, если в результирующем упорядоченном контейнере относительный порядок расположения элементов с равными ключами такой же, что и в исходном. Неустойчивым алгоритмом называется алгоритм сортировки, который нарушает порядок взаимного расположения элементов с равными значениями ключей.

Простейшим алгоритмом сортировки является алгоритм пузырьковой сортировки (*bubble sort*). При выполнении этого алгоритма контейнер с данными просматривается поэлементно, на каждом проходе элементы контейнера сравниваются попарно при помощи предиката сортировки. Если оказывается, что порядок элементов неверен, то элементы меняются местами. Алгоритм продолжает работу до тех пор, пока существуют неверно упорядоченные элементы. Если на очередном проходе таких элементов найдено не будет, то алгоритм завершает работу, и контейнер оказывается упорядоченным. При работе алгоритм фактически находит элементы, стоящие не на «своем» месте, и затем перемещает их в нужную позицию, т.е. образно говоря элемент «всплывает» до нужной позиции, как пузырек в воде.

2. Приведем описание алгоритма в словесно-формульном виде. В качестве контейнера данных будем использовать массив.

Пузырьковая сортировка

Входные данные: $A = a_0, a_1, a_2, \ldots, a_{N-1}$ – сортируемый массив;

$Op(x, y)$ – бинарный предикат упорядочивания, если $Op(a_i, a_j) = true$, тогда считается что элементы $a_i$ и $a_j$ упорядочены верно;

Промежуточные данные: *i, j* – счетчики обработанных элементов

Выходные данные: *A* – массив, для которого $Op(a_i, a_j) = true$ при любых *i, j*

1. Инициализируем счетчик упорядоченных элементов *i* индексом последнего элемента *i = N–1*, т.е. считаем, что еще ни один элемент не упорядочен
2. Проверяем, все ли элементы упорядочены (*i = 0*), если да, то завершаем работу
3. Начиная с первого элемента массива просматриваем массив для нахождения неправильно расположенных элементов, для этого инициализируем счетчик просматриваемых элементов *j = 0*
4. Если просмотрели еще не все неупорядоченные элементы (*j < i*), проверяем, правильно ли упорядочена пара элементов $a_j$ и $a_{j+1}$, для этого проверяем предикат $Op\ (a_j, a_{j+1})$, если элементы упорядочены неправильно, т.е. $Op\ (a_j, a_{j+1}) = false$, то меняем элементы $a_j$ и $a_{j+1}$местами
5. Переходим к следующему элементу, увеличивая счетчик *j = j + 1*, и переходим к шагу 4
6. Уменьшаем счетчик *i*, увеличивая количество уже упорядоченных элементов *i = i – 1*, и повторяем действия алгоритма, начиная с шага 2

3. Оценку времени выполнения будем проводить тестовой программой

```
type
TBinaryPredicate = function (x, y: Integer): Boolean;
function Greater (x, y: Integer): Boolean;
begin
Result:= x > y;
end;
function Lower (x, y: Integer): Boolean;
begin
Result:= x < y;
end;
procedure BubbleSort (var A: array of Integer; Op: TBinaryPredicate);
var
i, j: Integer;
t: Integer;
begin
for i:= High (A) downto Low (A) do
for j:= Low (A) to i - 1 do
if not Op (A[j], A[j + 1]) then
begin
t:= A[j];
A[j]:= A[j + 1];
A[j + 1]:= t;
end;

end;
```

Программа запускает серию тестов для разных размеров массива A, замеряет время выполнения процедуры BubbleSort для трех случаев входных данных – общий случай, массив заполнен случайными числами, то есть числами в случайном порядке, случае, когда массив уже отсортирован, и случае, когда массив отсортирован в обратном порядке. Для измерения времени выполнения алгоритма в тестовой программе используются функции ОС Windows QueryPerformanceCounter и QueryPerformanceFrequency. Первая функция получает текущее значение высокоточного

таймера, вторая функция – частоту. Для измерения времени выполнения алгоритма необходимо получить значение таймера до и после выполнения, а затем разность этих значений разделить на частоту таймера. Каждое измерение проводится пять раз, затем полученные значения усредняются. Это сделано для уменьшения влияния на измерения внешних факторов. По результатам измерений построим таблицу и график времени работы от размера массива.

Таблица 1

| **Размер массива N** | **Общий случай, мс (General)** | **Уже сортирован, мс (Sorted)** | **Сортирован в обратном порядке, мс (Rev.Sorted)** |
|---|---|---|---|
| 100 | 0,05 | 0,04 | 0,04 |
| 500 | 1,57 | 1,07 | 1,19 |
| 1000 | 6,67 | 5,99 | 5,13 |
| 1500 | 15,36 | 11,81 | 11,92 |
| 2000 | 31,21 | 21,95 | 30,08 |
| 5000 | 208,96 | 114,95 | 135,22 |
| 10000 | 635,58 | 480,65 | 649,38 |
| 50000 | 20983,63 | 12646,30 | 19328,59 |

Таким образом, время выполнения алгоритма пузырьковой сортировки зависит от квадрата количества элементов сортируемого массива. Алгоритм пузырьковой сортировки является одним из самых медленных алгоритмов сортировки.

Задание**:**

Написать программу, реализующую алгоритмы обхода графа «в глубину» и «в ширину».

Порядок выполнения:

При поиске в глубину посещается первая вершина, затем необходимо идти вдоль ребер графа, до попадания в тупик. Вершина графа является тупиком, если все смежные с ней вершины уже посещены. После попадания в тупик нужно возвращаться назад вдоль пройденного пути, пока не будет обнаружена вершина, у которой есть еще не посещенная вершина, а затем необходимо двигаться в этом новом направлении. Процесс оказывается завершенным при возвращении в начальную вершину, причем все смежные с ней вершины уже должны быть посещены.

Таким образом, основная идея поиска в глубину – когда возможные пути по ребрам, выходящим из вершин, разветвляются, нужно сначала полностью исследовать одну ветку и только потом переходить к другим веткам (если они останутся нерассмотренными).

Алгоритм поиска в глубину

Шаг 1. Всем вершинам графа присваивается значение не посещенная. Выбирается первая вершина и помечается как посещенная.

Шаг 2. Для последней помеченной как посещенная вершины выбирается смежная вершина, являющаяся первой помеченной как не посещенная, и ей присваивается значение посещенная. Если таких вершин нет, то берется предыдущая помеченная вершина.

Шаг 3. Повторить шаг 2 до тех пор, пока все вершины не будут помечены как посещенные.

Демонстрация алгоритма поиска в глубину

```
//Описание функции алгоритма поиска в глубину
void Depth_First_Search(int n, int **Graph, bool *Visited,
int Node){
Visited[Node] = true;
cout << Node + 1 << endl;
for (int i = 0 ; i < n ; i++)
if (Graph[Node][i] && !Visited[i])
Depth_First_Search(n,Graph,Visited,i);
}
```

Также часто используется нерекурсивный алгоритм поиска в глубину. В этом случае рекурсия заменяется на стек. Как только вершина просмотрена, она помещается в стек, а использованной она становится, когда больше нет новых вершин, смежных с ней.

Временная сложность зависит от представления графа. Если применена матрица смежности, то временная сложность равна $O(n^2)$, а если нематричное представление – $O(n+m)$: рассматриваются все вершины и все ребра.

**Поиск в ширину**

При поиске в ширину, после посещения первой вершины, посещаются все соседние с ней вершины. Потом посещаются все вершины, находящиеся на расстоянии двух ребер от начальной. При каждом новом шаге посещаются вершины, расстояние от которых до начальной на единицу больше предыдущего. Чтобы предотвратить повторное посещение вершин, необходимо вести список посещенных вершин. Для хранения временных данных, необходимых для работы алгоритма, используется очередь – упорядоченная последовательность элементов, в которой новые элементы добавляются в конец, а старые удаляются из начала.

Таким образом, основная идея поиска в ширину заключается в том, что сначала исследуются все вершины, смежные с начальной вершиной (вершина с которой начинается обход). Эти вершины находятся на расстоянии 1 от начальной. Затем исследуются все вершины на расстоянии 2 от начальной, затем все на расстоянии 3 и т.д. Обратим внимание, что при этом для каждой вершины сразу находятся длина кратчайшего маршрута от начальной вершины.

Алгоритм поиска в ширину

Шаг 1. Всем вершинам графа присваивается значение не посещенная. Выбирается первая вершина и помечается как посещенная (и заносится в очередь).

Шаг 2. Посещается первая вершина из очереди (если она не помечена как посещенная). Все ее соседние вершины заносятся в очередь. После этого она удаляется из очереди.

Шаг 3. Повторяется шаг 2 до тех пор, пока очередь не пуста.

Демонстрация алгоритма поиска в ширину

```
//Описание функции алгоритма поиска в ширину
void Breadth_First_Search(int n, int **Graph, bool *Visited, int Node){
int *List = new int[n]; //очередь
int Count, Head; // указатели очереди
int i;
```

```
// начальная инициализация
for (i = 0; i < n ; i++)
List[i] = 0;
Count = Head = 0;
// помещение в очередь вершины Node
List[Count++] = Node;
Visited[Node] = true;
while (Head < Count) {
//взятие вершины из очереди
Node = List[Head++];
cout << Node + 1 << endl;
// просмотр всех вершин, связанных с вершиной Node
for (i = 0 ; i < n ; i++)
// если вершина ранее не просмотрена
if (Graph[Node][i] && !Visited[i]){
// заносим ее в очередь
List[Count++] = i;
Visited[i] = true;
}
}
}
```

Сложность поиска в ширину при нематричном представлении графа равна O(n+m), ибо рассматриваются все n вершин и m ребер. Использование матрицы смежности приводит к оценке $O(n^2)$

Форма отчетности:

Отчет по лабораторной работке должен содержать:

1. Наименование лабораторной работы;
2. Разработанную программу;
3. Результаты её тестирования;
4. Выводы по работе.

Задания для самостоятельной работы:

1. Произвести анализ и оценку времени работы алгоритма

1) Разложение числа на множители.
2) Проверка делимости натурального числа на 3.
3) Нахождение суммы элементов массива.
4) Вычисление факториала натурального числа.
5) Сложение и вычитание нескольких *n*-разрядных чисел по школьному правилу «столбиком».
6) Умножение *n*-разрядных чисел по школьному правилу «столбиком».
7) Перевод десятичного числа в двоичное.
8) Распознавание вхождения слова A в слово B.
9) Сложение двух матриц порядка *n*.
10) Произведение двух матриц порядка *n*.
11) Поиск элемента в упорядоченном одномерном массиве.
12) Поиск остатка от деления числа *a* на число *b*.
13) Произведение двух матриц порядка *n*.
14) Вычисление среднего значения в выборке.

2. Написать программу, состоящую из следующих процедур:

1. Процедура поиска гамильтоновых циклов алгоритмом Робертса –Флореса.
2. Процедура решения задачи коммивояжера методом ветвей и границ.
3. Процедура решения задачи коммивояжера методом ближайшего города или соседа.

4. Процедура решения задачи коммивояжера итерационным методом.

Рекомендации по выполнению заданий и подготовке к лабораторной работе

1. Ознакомиться с заданием;
2. Изучить теоретические сведения, полученные на лекции;
3. Ознакомиться с примерами решения подобных задач в учебной литературе;
4. Выполнить задание.

Основная литература

1. Незнанов, А. А. Программирование и алгоритмизация : учебник / А. А. Незнанов. - Москва : Академия, 2010. - 304 с.
2. Ратинская, Е. В. Теория алгоритмов : учебное пособие / Е. В. Ратинская. - Братск : БрГУ, 2011. - 83 с.

Контрольные вопросы для самопроверки

1. Как связаны между собой различные способы представления графов?
2. Как от вида или представления графа зависит временная сложность алгоритмов поиска в глубину и в ширину?
3. В чем отличие разрешающего алгоритма от перечисляющего?
4. Верно ли что всякое разрешимое множество – перечислимо? Верно ли обратное?

## 10. ПЕРЕЧЕНЬ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, ИСПОЛЬЗУЕМЫХ ПРИ ОСУЩЕСТВЛЕНИИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

1. Microsoft Imagine Premium: Microsoft Windows Professional 7;
2. Microsoft Office 2007 Russian Academic OPEN No Level;
3. Антивирусное программное обеспечение Kaspersky Security;
4. GNU gcc;
5. ОС Linux;
6. LibreOffice.

## 11. ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

| ***Вид занятия*** | ***Наименование аудитории*** | ***Перечень основного оборудования*** | ***№ ЛР,ПЗ*** |
|---|---|---|---|
| **1** | **2** | **3** | **4** |
| Лк | Лекционная аудитория | - | - |
| ЛР | Лаборатория параллельных вычислений | Персональные компьютеры i5-2500/H67/4Gb/500Gb (монитор TFT19 Samsung E1920NR); интерактивная доска Smart Board X885ix со встроенным проектором UX60 | № 1-2 |
| ПЗ | Лекционная аудитория | - | № 1-2 |
| СР | ЧЗ1 | Оборудование 10 ПК i5-2500/H67/4Gb(монитор TFT19 Samsung); принтер HP LaserJet P2055D | - |

**Приложение 1**

# ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

## 1. Описание фонда оценочных средств (паспорт)

| № компетенции | Элемент компетенции | Раздел | Тема | ФОС |
|---|---|---|---|---|
| **ОПК-1** | Способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой | **1**. Математические модели алгоритмов | 1.1. Основные понятия теории алгоритмов | Индивидуальное задание, экзаменационный вопрос |
| | | | 1.2. Элементы формальной грамматики | Индивидуальное задание, экзаменационный вопрос |
| **ОПК-3** | Способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей | | 1.3. Математические модели алгоритма | Индивидуальное задание, экзаменационный вопрос |
| | | | 1.4. Алгоритмические проблемы | Индивидуальное задание, экзаменационный вопрос |
| **ПК-7** | Способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения | **2**. Анализ эффективности алгоритма | 2.1 Введение в теорию сложности | Индивидуальное задание, экзаменационный вопрос |
| | | | 2.2 Приближенные алгоритмы и эвристики | Индивидуальное задание, экзаменационный вопрос |

**2. Экзаменационные вопросы**

| № п/п | Компетенции | | ЭКЗАМЕНАЦИОННЫЕ ВОПРОСЫ | № и наименование раздела |
|---|---|---|---|---|
| | Код | Определение | | |
| 1 | 2 | 3 | 4 | 5 |
| **1.** | ОПК-1 | Способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой | 1. Предмет изучения теории алгоритмов. Алгоритм и его свойства. Массовая задача. | **1**. Математические модели алгоритмов |
| | | | 2. Способы описания алгоритма. Примеры алгоритмов. | **1.** Математические модели алгоритмов |
| | | | 3. Регулярные языки. Грамматики. | **1**. Математические модели алгоритмов |
| | | | 4. Конечные автоматы. | **1**. Математические модели алгоритмов |
| **2.** | ОПК-3 | Способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей | 5. Машина Тьюринга. | **1.** Математические модели алгоритмов |
| | | | 6. Понятие рекурсивной функции. Простейшие функции. Операции над функциями.. | **1**. Математические модели алгоритмов |
| | | | 7. Марковские подстановки. | **1**. Математические модели алгоритмов |
| | | | 8. Проблема распознавания применимости и самоприменимости. | **1.** Математические модели алгоритмов |
| **3.** | ПК-7 | Способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения | 9. Виды эффективности алгоритма. Экспериментальная оценка сложности. | **2.** Анализ эффективности алгоритма |
| | | | 10. Теоремы о соотношении пространственной и временной эффективности. | **2.** Анализ эффективности алгоритма |
| | | | 11. Сложность задачи. Классы асимптотической сложности. | **2.** Анализ эффективности алгоритма |
| | | | 12. Примеры NP- сложных задач . Основная проблема теории сложности. | **2.** Анализ эффективности алгоритма |
| | | | 13. Полиномиальная сводимость. Задача оптимизации. | **2.** Анализ эффективности алгоритма |
| | | | 14. Приближенные алгоритмы. Мультипликативная ошибка. Эвристики. | **2.** Анализ эффективности алгоритма |

### 3. Описание показателей и критериев оценивания компетенций

| Показатели | Оценка | Критерии |
|---|---|---|
| **знать:**<br>ОПК-1<br>- основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей;<br>ОПК-3<br>- алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>ПК-7<br>- основные понятия теории алгоритмов, определения и свойства математических объектов, используемых в этой области, формулировки утверждений, методы их доказательства, возможные сферы их приложений;;<br>**уметь:**<br>ОПК-1<br>- формализовать задачу и описать ее с помощью известных математических моделей;<br>ОПК-3<br>- разрабатывать программные решения для обработки и анализа моделей;<br>ПК-7<br>- решать задачи теоретического и прикладного характера из различных разделов теории алгоритмов; | **отлично** | Демонстрирует все показатели компетенций на высоком уровне, а именно:<br>знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>-владеет методами и приемами обработки данных и интерпретации результатов;<br>Владеет способами обработки, анализа и систематизации информации с использованием полученных знаний. |
| | **хорошо** | Демонстрирует освоенность не менее 8 показателей компетенций:<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных |

| | | |
|---|---|---|
| **владеть:**<br>ОПК-1<br>- навыками применения аппарата математики и информатики для решения прикладных задач<br>ОПК-3<br>– методами и приемами обработки данных и интерпретации результатов<br>ПК-7<br>– навыками алгоритмизации основных задач. | | математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>-владеет методами и приемами обработки данных и интерпретации результатов;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>-владеет способами обработки, анализа и систематизации информации с использованием полученных знаний ;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации; |

| | | |
|---|---|---|
| | | -умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет методами и приемами обработки данных и интерпретации результатов;<br>-владеет способами обработки, анализа и систематизации информации с использованием полученных знаний. |
| | **удовлетворительно** | Демонстрирует владение не менее, чем половиной (5) параметров компетенций:<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет применять математический |

| | | |
|---|---|---|
| | | аппарат для обработки данных;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-владеет методами и приемами обработки данных и интерпретации результатов<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и |

| | | |
|---|---|---|
| | | описать ее с помощью известных математических моделей;<br>-умеет применять математическ<sup></sup>ий аппарат для обработки данных. |
| | **неудовлетворительно** | Владеет менее чем половиной параметров компетенций. |

## 4 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и опыта деятельности

Дисциплина Теория алгоритмов направлена на ознакомление обучающихся с общими свойствами алгоритмов и их формальными моделями. К задачам теории алгоритмов относятся: доказательство алгоритмической неразрешимости задач, анализ трудности задач, анализ сложности и эффективности алгоритмов.

Изучение дисциплины Теория алгоритмов предусматривает:

- лекции,
- лабораторные работы;
- практические занятия;
- экзамен;
- самостоятельную работу.

Для фиксирования успешности обучения предусматривается экзамен.

В ходе освоения раздела 1 «Математические модели алгоритмов» обучающиеся должны изучить математические основы описания алгоритма, основные модели алгоритма, теоретические проблемы, связанные с возможностью построения алгоритмов решения математических задач.

В ходе освоения раздела 2 «Анализ эффективности алгоритма» обучающиеся осваивают понятия сложности и эффективности алгоритма, знакомятся с классами сложности задач, осваивают приемы приближенного решения труднорешаемых задач.

Обучающимся необходимо овладеть навыками и умениями применения изученных методов для разработки и реализации профессионально ориентированных проектов в последующей учебной деятельности.

Овладение ключевыми понятиями является основой усвоения учебного материала по дисциплине.

При подготовке к экзамену особое внимание необходимо уделить рекомендациям и замечаниям преподавателей, ведущих аудиторные занятия по дисциплине

В процессе проведения практических занятий происходит закрепление знаний, формирование умений и навыков применения различных методов решения стандартных математических ситуаций.

Самостоятельную работу необходимо начинать с чтения лекций и учебников.

В процессе консультации с преподавателем обучающийся выясняет наличие пробелов в знаниях и способах решения разных ситуаций.

Работа с литературой является важнейшим элементом в получении знаний по дисциплине. Прежде всего, необходимо воспользоваться списком рекомендуемой по данной дисциплине литературой. Дополнительные сведения по изучаемым темам можно найти в периодической печати и Интернете.

Предусмотрено проведение аудиторных занятий в виде разнообразных тренингов и ситуаций общения в сочетании с внеаудиторной работой.

**Приложение 2**

# АННОТАЦИЯ
# рабочей программы дисциплины
# Теория алгоритмов

**1. Цель и задачи дисциплины**

Целью изучения дисциплины является: ознакомление обучающихся с математическими моделями алгоритмов, основами теории сложности вычислений, привить практические навыки конструирования алгоритмов решения распространенных математических задач.

Задачами дисциплины являются

- формирование математической культуры обучающегося;
- обучение рациональным способам решения задач;
- формирование и развитие умений и навыков, позволяющих применять современные математические методы и программное обеспечение для решения задач науки и техники.

**2. Структура дисциплины**

2.1 Распределение трудоемкости по отдельным видам учебных занятий, включая самостоятельную работу: Лк.- 18 час., ЛР- 18 час., ПЗ-18 час., СР - 54 час.

Общая трудоемкость дисциплины составляет 144 часа, 4 зачетных единиц.

2.2 Основные разделы дисциплины:

1 –Математические модели алгоритмов;
2 –Анализ эффективности алгоритма.

**3. Планируемые результаты обучения (перечень компетенций)**

Процесс изучения дисциплины направлен на формирование следующих компетенций:

ОПК-1 Способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой

ОПК-3 Способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям

ПК-7 Способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения .

**4. Вид промежуточной аттестации**: экзамен.

**Приложение 3**

***Протокол о дополнениях и изменениях в рабочей программе***
***на 20___-20___ учебный год***

**1.** В рабочую программу по дисциплине вносятся следующие дополнения:
_____
_____

**2.** В рабочую программу по дисциплине вносятся следующие изменения:
_____
_____
_____

Протокол заседания кафедры №____ от «___» _____ 20 ____ г.,
*(разработчик)*

Заведующий кафедрой _____ _____
*(подпись)* *(Ф.И.О.)*

**Приложение 4**

# ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ ПО ДИСЦИПЛИНЕ

## 1. Описание фонда оценочных средств (паспорт)

| № компетенции | Элемент компетенции | Раздел | Тема | ФОС |
|---|---|---|---|---|
| **ОПК-1** | Способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой | **1**. Математические модели алгоритмов | 1.1. Основные понятия теории алгоритмов | ЛР№1 |
| **ОПК-3** | Способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей | | 1.3. Математические модели алгоритма | ЛР№1 |
| **ПК-7** | Способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения | **2**. Анализ эффективности алгоритма | 2.1 Введение в теорию сложности | ЛР№2 |
| | | | 2.2 Приближенные алгоритмы и эвристики | ЛР№2 |

| Показатели | Оценка | Критерии |
|---|---|---|
| **знать:**<br>ОПК-1<br>- основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей; | **отлично** | Демонстрирует все показатели компетенций на высоком уровне, а именно:<br>знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для |

| | | |
|---|---|---|
| ОПК-3<br>- алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>ПК-7<br>- основные понятия теории алгоритмов, определения и свойства математических объектов, используемых в этой области, формулировки утверждений, методы их доказательства, возможные сферы их приложений;;<br>**уметь:**<br>ОПК-1<br>- формализовать задачу и описать ее с помощью известных математических моделей;<br>ОПК-3<br>- разрабатывать программные решения для обработки и анализа моделей;<br>ПК-7<br>- решать задачи теоретического и прикладного характера из различных разделов теории алгоритмов;<br>**владеть:**<br>ОПК-1<br>- навыками применения аппарата математики и информатики для решения прикладных задач<br>ОПК-3<br>– методами и приемами обработки данных и интерпретации результатов<br>ПК-7<br>– навыками алгоритмизации | | обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>-владеет методами и приемами обработки данных и интерпретации результатов;<br>Владеет способами обработки, анализа и систематизации информации с использованием полученных знаний. |
| | **хорошо** | Демонстрирует освоенность не менее 8 показателей компетенций:<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>-владеет методами и приемами обработки данных и интерпретации результатов;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при |

| | | |
|---|---|---|
| основных задач. | | моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>-владеет способами обработки, анализа и систематизации информации с использованием полученных знаний ;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных;<br>-владеет методами и приемами обработки данных и интерпретации результатов;<br>-владеет способами обработки, анализа и систематизации информации с использованием полученных знаний . |

| | | |
|---|---|---|
| | **удовлетворительно** | Демонстрирует владение не менее, чем половиной (5) параметров компетенций:<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и описать ее с помощью известных математических моделей;<br>-умеет применять математический аппарат для обработки данных;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет формализовать задачу и |

| | | |
|---|---|---|
| | | описать ее с помощью известных математических моделей;<br>-владеет навыками применения аппарата математики и информатики для решения прикладных задач;<br>или<br>-знает основные понятия прикладной математики и информатики, законы естественных наук, применяемые при моделировании, способы построения математических моделей ;<br>-знает алгоритмы разработки информационных и имитационных моделей, основы программирования;<br>-знает - способы применения математического моделирования для обработки, анализа и систематизации информации;<br>-умеет разрабатывать программные решения для обработки и анализа моделей;<br>-умеет применять математический аппарат для обработки данных |
| | **неудовлетворительно** | Владеет менее чем половиной параметров компетенций. |

Программа составлена в соответствии с федеральным государственным образовательным стандартом высшего образования по направлению подготовки 01.03.02 Прикладная математика и информатика от «12» марта 2015 г. № 228

**для набора 2018 года** и учебным планом ФГБОУ ВО «БрГУ» для очной формы обучения от «12» марта 2018г. №130

**Программу составили:**

Багинова Т.Г. , к.т.н, доцент каф. МиФ _____

Ратинская Е.В., ст. препод. каф. МиФ _____

Рабочая программа рассмотрена и утверждена на заседании кафедры МиФ

от «21» ноября 2018 г., протокол № 3

И.о. зав.выпускающей кафедрой _____ О.И.Медведева

СОГЛАСОВАНО:

И.о. зав.выпускающей кафедрой _____ О.И Медведева.

Директор библиотеки _____ Т.Ф.Сотник

Рабочая программа одобрена методической комиссией ЕН факультета

от «20 » декабря 2018 г., протокол № 4

Председатель методической комиссии факультета _____ М.А. Варданян

СОГЛАСОВАНО:

Начальник
учебно-методического управления _____ Г.П. Нежевец

Регистрационный №_____