Приложение № 1
к приказу от 28 октября 2022 г. № 613

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

ПРОГРАММА
ВСТУПИТЕЛЬНОГО МЕЖДИСЦИПЛИНАРНОГО ЭКЗАМЕНА
В МАГИСТРАТУРУ ПО НАПРАВЛЕНИЮ ПОДГОТОВКИ
01.04.02 «ПРИКЛАДНАЯ МАТЕМАТИКА И ИНФОРМАТИКА»

Введение

Программа составлена в соответствии с самостоятельно устанавливаемым образовательным стандартом Московского авиационного института (национального исследовательского университета) по высшему профессиональному образованию по направлению подготовки 01.04.02 - Прикладная математика и информатика.

Цели и задачи вступительных испытаний

Вступительные испытания предназначены для определения практической и теоретической подготовленности бакалавра и проводятся с целью определения соответствия знаний, умений и навыков студентов требованиям обучения в магистратуре по направлению 01.04.02 - Прикладная математика и информатика.

Содержание вступительных испытаний

В соответствии с содержанием магистерской программы вступительные испытания в магистратуру по направлению подготовки 01.04.02 - Прикладная математика и информатика проводятся по следующим дисциплинам: фундаментальная информатика, вычислительные системы, дискретный анализ, языки и методы программирования, практикум на ЭВМ, объектно-ориентированное программирование.

Вступительные испытания в магистратуру по направлению подготовки 01.04.02 - Прикладная математика и информатика проводятся по следующим критериям:

– оценка соответствия профиля и уровня полученного образования;

– подготовленность к научно-исследовательской работе.

Оценка уровня знаний

Оценка уровня знаний проводится в виде вступительного экзамена. В основу программы вступительного экзамена положены квалификационные требования в области информатики, предъявляемые к бакалаврам направления 01.04.02 - Прикладная математика и информатика.

Вступительный экзамен проводится в письменной форме. Применение ЭВМ на экзамене возможно в соответствии с утверждённым регламентом.

Оценки выставляются по стобалльной шкале. Оценки 30 – 100 являются положительными.

На итоговую оценку влияют: полнота и безошибочность ответов на теоретические вопросы, наличие обоснованного решения задачи с анализом возможных алгоритмов её решения и с качественным программным кодом и тестами. При равенстве баллов приёмная комиссия принимает во внимание наличие:

1) рекомендации ГЭК в магистратуру;

2) диплома с отличием по результатам освоения профильной программы бакалавриата;

3) дипломов, сертификатов и грамот за успешные выступления на олимпиадах по программированию, конкурсах студенческих проектов в IT-сфере;

4) результатов научной работы по профилю магистерской программы: публикаций, докладов на конференциях, изобретений, зарегистрированных программ для ЭВМ;

5) стажа работы в высокотехнологичных IT-компаниях, научно-исследовательских и опытно-конструкторских организациях на профильных должностях.

Перечень вопросов для вступительных испытаний в магистратуру по направлению 01.04.02-Прикладная математика и информатика

На экзамене по данным дисциплинам могут быть предложены как теоретические вопросы, так и задачи на программирование в том числе и с отладкой программ на ЭВМ по следующим темам:

Фундаментальная информатика

1. Предмет информатики. Информация и сообщения. Интерпретация сообщений. Знаки и символы. Кодирование. Системы счисления.
2. Обработка сообщений. Обработка информации. Автоматизация обработки информации. Конструктивное описание процесса обработки дискретных сообщений.
3. Понятие алгоритма. Свойства алгоритмов. Сложность алгоритмов. Необходимость формального определения алгоритма.
4. Семиотические модели интерпретации дискретных сообщений.
5. Машины Тьюринга. Нормальные алгоритмы Маркова.
6. Диаграммы машин Тьюринга.
7. Моделирование машин Тьюринга.

8. Эквивалентность программ и диаграмм, диаграмм и программ.

9. Теоремы Шеннона о машинах Тьюринга с двумя внутренними состояниями и с однобуквенным рабочим алфавитом.

10. Вычислимые функции. Нормированные вычисления. Теорема о нормированной вычислимости.

11. Теоремы о сочетаниях алгоритмов (композиция, ветвление, цикл).

12. Схемы машин Тьюринга. Нисходящая разработка. Теорема Бойма-Джакопини-Миллса.

13. Универсальная машина Тьюринга.

14. Линейная запись схем машин Тьюринга.

15. Критика модели вычислений Тьюринга.

16. Алгоритмическая модель фон Неймана. Адреса и имена.

17. Специализированные процессоры для обработки сообщений.

18. Построение универсального процессора фон Неймана.

19. Машина фон Неймана: принципы реализации.

20. Структура программ для машины фон Неймана.

21. Нотация программ Дейкстры. Обобщенные инструкция присваивания, композиции, ветвления и цикла.

22. Понятие типа данных.

23. Тип логический.

24. Тип литерный.

25. Тип целый.

26. Тип вещественный.

27. Согласование типов.

28. Небазовые типы данных (диапазон, перечисление, множество).

29. Структурные типы данных. Тип массив, тип запись и тип файл.

30. Блочная структура программ. Локальные и глобальные переменные.

31. Процедуры и функции. Описание, вызов, способы передачи параметров.

32. Понятие рекурсии. Рекурсия и итерация. Рекурсивный вызов процедур.

33. Критика алгоритмической модели фон Неймана.

Языки и методы программирования

1. Статические и динамические объекты программ. Ссылочный тип данных.

2. Файл. Функциональная спецификация. Логическое описание. Физическое представление.

3. Вектор. Функциональная спецификация. Логическое описание и физическое представление.

4. Очередь. Функциональная спецификация. Логическое описание и физическое представление.

5. Стек. Функциональная спецификация. Логическое описание. Физическое представление.

6. Дек. Сравнительное описание. Примеры задач.

7. Линейный список. Функциональная спецификация. Логическое описание. Физическое представление. Итераторы.

8. Списки общего вида. Представление и обработка графов.

9. Деревья. Двоичные деревья.

10. Двоичное дерево. Функциональная спецификация. Логическое описание. Физическое представление.

11. Двоичное дерево. Построение и визуализация.

12. Алгоритмы обхода деревьев. Прошивка.

13. Особенности представления и обработки деревьев общего вида. Преобразование к двоичным деревьям.

14. Деревья выражений. Алгоритм Рутисхаузера. Алгоритм Бауэра-Замельзона. Алгоритм Дейкстры.

15. Деревья поиска. Сбалансированные деревья поиска. AVL-деревья, вставка и удаление элемента.

16. Линейный поиск. Поиск по образцу в последовательностях и таблицах. Алгоритм Кнута-Морриса-Пратта. Алгоритм Бойера-Мура. Алгоритм Рабина-Карпа.

17. Таблицы с прямым доступом.

18. Простые методы сортировки. Сортировка Шелла.

19. Усовершенствованные методы сортировки. Турнирные сортировки. Гладкая сортировка. Сортировка Хоора. Сортировки слиянием.

Дискретный анализ

1. Недостижимость линейной оценки времени для алгоритмов сортировок, основанных на сравнениях элементов.

2. Сортировки за линейное время. Сортировка подсчётом.

3. Сортировки за линейное время. Поразрядная сортировка.

4. Сортировки за линейное время. Карманная сортировка.

5. Деревья поиска. Сбалансированные деревья. Красно-черные деревья. Вставка нового узла в красно-черное дерево.

6. Деревья поиска. Сбалансированные деревья. Красно-черные деревья. Удаление узла из красно-черного дерева.

7. Сильноветвящиеся деревья, B-деревья. Вставка элемента в B-дерево.

8. Сильноветвящиеся деревья, B-деревья. Удаление элемента из B-дерева.

9. Рандомизированные деревья поиска. Декартовы деревья. Вставка нового узла в декартово дерево.

10. Рандомизированные деревья поиска. Декартовы деревья. Удаление узла из декартова дерева.

11. Дерево ключей. Дерево PATRICIA. Вставка нового элемента в дерево PATRICIA.

12. Дерево ключей. Дерево PATRICIA. Удаление элемента из дерева PATRICIA.

13. Поиск образца в строке. Основной алгоритм предварительной обработки образца. Использование Z-блоков для поиска образца в строке.

14. Алгоритм Кнута-Мориса-Пратта. Построение функций sp и sp' на основе Z-блоков. Вариант алгоритма Кнута-Мориса-Пратта реального времени.

15. Алгоритм Кнута-Мориса-Пратта. Построение функций sp и sp' «классическим» способом.

16. Множественный поиск образцов в строке. Алгоритм Ахо-Корасик.

17. Построение связей неудач для дерева ключей. Поиск в строке образца с метасимволами («джокерами»).

18. Алгоритм Бойера-Мура. Правило плохого символа, расширенное правило плохого символа. Предварительная обработка образца с использованием Z-блоков. Реализация алгоритма Бойера-Мура.

19. Алгоритм Апостолико-Джанкарло с линейной оценкой времени.

20. Суффиксные деревья. Наивный алгоритм построения суффиксного дерева. Примеры использования суффиксных деревьев.

21. Обобщенное суффиксное дерево для набора строк. Трудности использования суффиксных деревьев для алфавитов большой размерности.

22. Суффиксные деревья. Приложения суффиксных деревьев: поиск образца в строке, множественный поиск образцов в строке.

23. Суффиксные деревья. Приложения суффиксных деревьев: Решение задачи о наибольшей общей подстроке для двух строк; для большого количества строк; линеариазация циклической строки.

24. Суффиксные массивы. Бинарный поиск образца в суффиксном массиве. Уменьшение времени поиска до O(n + logm).

25. Суффиксные массивы. Алгоритм построения за O(n logn) без использования суффиксных деревьев.

26. Длинная арифметика. Классические алгоритмы сложения, вычитания, умножения и возведения в степень длинных чисел.

27. Длинная арифметика. Классический алгоритм деления длинных чисел.

28. Быстрое умножение длинных чисел: алгоритм Карацубы и его сложность.

29. Полиномы, коэффициентное и интерполяционное представление полиномов. Операции над полиномами. Дискретное преобразование Фурье.

30. Алгоритм быстрого преобразования Фурье. Прямое и обратное БПФ. Использование БПФ для перемножения полиномов.

31. Динамическое программирование. Перекрытие подзадач, построение оптимального решения. Применение динамического программирования для решения задачи о расписании работы конвейера.

32. Динамическое программирование. Перекрытие подзадач, построение оптимального решения. Применение динамического программирования для решения задачи о перемножении цепочки матриц.

33. Динамическое программирование. Перекрытие подзадач, построение оптимального решения. Задача о максимальном совпадении двух строк и поиске самой длинной общей подпоследовательности. Расстояние Левенштейна.

34. Жадные алгоритмы, связь с динамическим программированием. Свойство жадного выбора и элементы жадной стратегии. Решение задачи о выборе процессов.

35. Жадные алгоритмы. Свойство жадного выбора и элементы жадной стратегии. Понятие о префиксных кодах, коды Хаффмана, их построение. Корректность алгоритма Хаффмана.

36. Жадные алгоритмы. Поиск максимальной возрастающей подпоследовательности.

37. Жадные алгоритмы. Сведение задачи о максимальной совпадающей подпоследовательности к задаче о максимальной возрастающей подпоследовательности. Максимальная совпадающая подпоследовательность более чем для двух строк.

38. Наивный байесовский классификатор.

39. Коды Хаффмана. Неоднозначность кодов Хаффмана. Адаптивные коды Хаффмана.

40. Канонические коды Хаффмана. Эффективное кодирование и декодирование с использованием канонических кодов Хаффмана.

41. Канонические коды Хаффмана. Быстрое вычисление длин кодов Хаффмана.

42. Арифметическое кодирование. Реализация с использование чисел фиксированной точности.

43. Символьные модели сжатия текстов. Преобразование Барроуза-Уиллера. Использование преобразования Барроуза-Уиллера для сжатия текстов, связь с суффиксными деревьями. Кодирование методом Move-To-Front.

44. Символьные модели сжатия текстов. Предсказание по частичному совпадению, различные варианты метода.

45. Словарные методы сжатия. Алгоритм LZ77. Реализация LZ77 в программе gzip.

46. Словарные методы сжатия. Алгоритм LZ77. Реализация LZ77 с использованием суффиксных деревьев.

47. Суффиксные деревья. Алгоритм Укконена. Использование суффиксных связей и прыжка по счетчику. Ускорение с O(n^3) до O(n^2).

48. Суффиксные деревья. Алгоритм Укконена. Сжатие дуговых меток, ускорение до O(n). Доказательство линейности работы.

Объектно-ориентированное программирование.

1. Ручное управление памятью. Выделение памяти на стеке. Опасности при возвращении адресов локальных переменных. Объект и его свойства. Классы объектов. Пространства имен. Жизненный цикл классов. Конструкторы и деструкторы.

2. Понятие объектно-ориентированного языка. Объектно-ориентированная декомпозиция. Базовые понятия ООП: абстрагирование. Абстракция сущности, поведения, виртуальной-машины, произвольная. Работа с вводом/выводом в C++. Потоки вывода и ввода.

3. Ссылки. Отличие от указателей. Lvalue и RValue переменные. LValue и RValue ссылки. Указатель nullptr. Базовые понятия ООП: инкапсуляция в C++.

4. Базовые понятия ООП: модульность в C++. Базовые понятия ООП: иерархия в C++. Аггрегация и наследование. Виртуальные функции и полиморфизм. Модификаторы

override и final. Абстрактные классы. Работа конструкторов и деструкторов при наследовании.

5. Базовые понятия ООП: Типизация и эквивалентность типов в C++. Базовые понятия ООП: Сохраняемость. Временные объекты в C++. Модификатор const в C++. Модификатор mutable в C++. Модификатор constexpr в C++. Явные и неявные конструкторы. Ключевое слово explicit.

6. Перегрузка операторов. Синтаксис. Возможные к перегрузке операции. Перегрузка ++ и --. Перегрузка бинарных операторов.

7. Перегрузка оператора =. Запрет операторов копирования. Перегрузка (). Функтор. Пользовательские литералы.

8. Исключения в C++. std::exception и std::exception_ptr. std::current_exception и std::rethrow_exception. Модификатор noexcept. Метод terminate

9. Идиома RAII. Умные указатели. std::shared_ptr. Наследование и std::shared_ptr. Проблемы при использовании std::shared_ptr.

10. Идиома RAII. Умные указатели. Шаблон std::weak_ptr. Шаблон std::unique_ptr. Семантика перемещения. std::move

11. Шаблоны классов и функций. Сравнение наследования и шаблонов. Параметры шаблонов. Специализация шаблонов.

12. Шаблоны классов и функций. Вычисление факториала на шаблонах.

13. Шаблоны классов и функций. inclusion model и explicit instantiation model. Шаблоны с переменным числом параметров.

14. Шаблоны классов и функций. Устройство шаблона tuple

15. Шаблоны классов и функций. Couriously Recurring Template Pattern. Множественное наследование классов с помощью шаблонов.

16. Шаблоны классов и функций. Шаблон из шаблонов. Ключевое слово auto. template alias. Протечка абстракции.

17. Итераторы. range-based циклы. std::initializer_list

18. Контейнеры STL. vector. Итераторы.

19. Контейнеры STL. map. Итераторы.

20. Контейнеры STL. deque. Итераторы.

21. Контейнеры STL. set. Итераторы.

22. Контейнеры STL. bitset. Итераторы.

23. Аллокаторы памяти. Перегрузка оператора new. Простой аллокатор памяти на массиве.

24. Проектирование структуры классов. Характеристики (жесткость, хрупкость, повторное использование. Способы улучшения структуры классов. SOLID: Принцип единой ответственности. Пример.

25. Метрика cohesion. Метрика coupling. Виды связанности. SOLID: Принцип открытости/закрытости. Пример.

26. Шаблон проектирования template method (обычный и CRTP). Шаблон проектирования strategy.

27. SOLID: Принцип подстановки Барбары Лисков.

28. Принцип TDA. Принцип Command Query Separation.Закон Постеля.

29. SOLID: Принцип разделения интерфейсов.

30. SOLID: Dependency Inversion Principle.

31. Мультипроцессирование и мультипрограммирование. SMP, MPP и NUMA. Мультипрограммирование. Вытесняющая и не вытесняющая многозадачность. Планировщик задач. Жизненный цикл потока.

32. std::thread. Функции и функторы как параметры. Использование семантики перемещения в std::thread. Функциии из пространства имен std::this_thread.

33. Потокобезопасность. Реентерабельность. Race condition. Взаимное исключение. std::mutex и std::recursive_mutex.

34. std::lock_guard и std::unique_lock. Реализация потокобезопасного стека. dead_lock

35. Просачивание данных за пределы lock_guard. std::future и std::async std::promise.

36. Условные переменные. Закон Амдала.

37. Неблокирующие алгоритмы. Атомарные типы. CAS операции. Потокобезопасный стек на CAS-операциях.

38. Лямбда-выражения. Списки захвата в лямбда выражениях. Лямбда-выражения, порождающие лямбда-выражения.

6. Пример билета для вступительных испытаний в магистратуру

Московский авиационный институт (национальный исследовательский университет)
Вступительный междисциплинарный экзамен в магистратуру
по направлению подготовки
01.04.02 «Прикладная математика и информатика»

Билет № 0

1. Вопрос по дисциплине «Вычислительные системы». Системы счисления.

2. Вопрос по дисциплине «Объектно-ориентированное программирование». Язык C++. Контейнеры STL. vector. Итераторы.

3. Задача на программирование по дисциплине «Объектно-ориентированное программирование». Язык C++. Переопределение операций: реализуйте класс для комплексного числа (Complex). Перегрузите операцию вывода в поток std::ostream. Перегрузите операцию сложения комплексных чисел. Модифицируйте операцию, так что бы она не только складывала числа, но и меняла их значения местами.

4. Задача на программирование по дисциплине «Практикум на ЭВМ». Языки C, C++ или Java. Составить функцию определения глубины двоичного дерева.

5. Задача на программирование по дисциплине «Дискретный анализ». Языки C, C++ или Java.

Асе очень нравятся бинарные деревья и сегодня её крайне интересует следующий вопрос: каково количество различных бинарных деревьев из N вершин, так как их число может быть слишком велико.

Будучи чрезвычайно практичной девушкой, она попросила Дину запрограммировать данное вычисление и вывести результат по модулю M ($1 \le M \le 10^9+7$).

Входные данные

В единственной строке дано единственное число N ($0 \le N$).

Выходные данные

Выведите единственное число — количество бинарных деревьев из N вершин.

Пример входных данных

1

Пример выходных данных

1

Пример входных данных
3

Пример выходных данных
5

Временн*а*я сложность алгоритма: $O(N^2)$

6. Задача на программирование по дисциплине «Языки и методы программирования». Языки C, C++ или Java.

Также Асе очень нравятся строки, поэтому на каждый день рождения друзья дарят ей новые строки. За всё время у неё скопилась внушительная коллекция строк и она решила её проредить. Ася решила выкинуть все строки, которые содержатся в каких-либо других строках, так как они не добавляют разнообразия в коллекцию. Так как строки могут быть очень длинными, ей не очень хочется самой проверять, входит ли одна строка в другую, поэтому она попросила Дину написать программу, которая определит это.

Входные данные

В двух строках A и B даны строки из малых латинских букв.

Выходные данные

Если строка A встречается в строке B как подстрока, выведите “Yes”, в противном случае выведите “No” (без кавычек).

Пример входных данных
bcd
abxde

Пример выходных данных
No

Пример входных данных

a

xax

Пример выходных данных

Yes

Временн*ая* сложность алгоритма: O(|A|+|B|), где |S| — длина строки S.

7. Задача на программирование по дисциплинам «Дискретная математика» и «Дискретный анализ». Языки C, C++ или Java.

Помимо всего прочего, Ася обожает графы. Сегодня она решила скачать граф одной известной социальной сети, в котором вершины – это аккаунты пользователей, а рёбра – это отношения дружбы между ними, и посчитать для каждой его вершины Асино число. Числом Аси для вершины является минимальное количество ребер, которые нужно пройти из данной вершины, чтобы оказаться в вершине, соответствующей аккаунту Аси. Если же из некоторой вершины невозможно добраться до вершины Аси, то Асиным числом для неё объявляется -1. Так как Ася ничего не понимает в программировании, а граф очень большой, она попросила Дину написать для неё программу, которая посчитает Асины числа для всех вершин.

Входные данные

В первой строке даны два числа N и M, число аккаунтов в социальной сети и число отношений дружбы соответственно. В следующих M строках даны описания отношений дружбы в виде пар чисел U и V ($1 \le U, V \le N$). Аккаунт Аси имеет номер 1.

Выходные данные

В одной строке выведите N чисел: Асины числа для каждого аккаунта в социальной сети.

Пример входных данных

4 3

1 2

2 4

4 1

Пример выходных данных

0 1 -1 1

Временн*а*я сложность алгоритма: O(N + M)

Рекомендуемая литература

1. Гайсарян С.С., Зайцев В.Е. Курс информатики. –М.: МАИ, 1993. (www.faq8.ru)
2. Нефедов В.Н., Осипова В.А. Курс дискретной математики: Учеб. пособие. – М.: 1992. –264 с.: ил. ISBN 5-7035-0157-X.
3. Ф.А. Новиков. Дискретная математика для программистов. – СПб: Изд. Питер, 2002.
4. Р.Седжвик. Фундаментальные алгоритмы на С++. Анализ/Структуры данных/Сортировка/Поиск: Пер. с англ./Роберт Седжвик. –СПб.: ООО ДиаСофтЮП, 2002. –688с.
5. Т.Кормен, Ч.Лейзерсон, Р.Ривест, Штайн К. Алгоритмы: построение и анализ. 2-е издание. Пер. с англ. –М.: «Вильямс», 2007. –1296с., ил. ISBN 5-8459-0857-4 (рус.).
6. М. Брой. Информатика. Основополагающее введение. В 4-х частях. –М.: Диалог-МИФИ, 1996.
7. Э. Танненбаум. Многоуровневая организация ЭВМ. –М.: Мир, 1979.
8. М. Брой. Информатика. Структуры систем и системное программирование. Ч.3. –М.: Диалог-МИФИ, 1996. -- 224с.
9. Г. Буч. Объектно-ориентированное проектирование с примерами применения. –М.: Конкорд, 1992. –519с., ил.
10. Б. Страуструп. Программирование. Принципы и практика с использованием С++. –М.: Вильямс, 2016. –1328с., ил. ISBN 978-5-8459-1949-6, 978-0-321-99278-9.
11. Генри С. Уоррен. Алгоритмические трюки для программистов. Пер. с англ. – М.: «Вильямс», 2004. –288с., ил.
12. Р. Грэхем, Д. Кнут, О. Паташник. Конкретная математика. Основание информатики. –М.: Мир, 1998.
13. Р. Хэзфилд, Л. Кирби и др. Искусство программирования на С. Фундаментальные алгоритмы, структуры данных и примеры приложений. Энциклопедия программиста: Пер. с англ. –К.: «ДиаСофт», 2001. – 736~с.
14. Д. Гасфилд. Строки, деревья и последовательности в алгоритмах: Информатика и вычислительная биология. –СПб: Невский Диалект, БХВ-Петербург, 2003. –654с, ил.