

**Паспорт Образовательной программы**

**«Разработчик JAVA. Продолжающий уровень»**

|  |  |
| --- | --- |
| **Версия программы** | **1** |
| **Дата Версии** | 08.10.2020 |

1. **Сведения о Провайдере**

|  |  |  |
| --- | --- | --- |
| 1.1 | Провайдер | АНО ВО "Университет Иннополис" |
| 1.2 | Логотип образовательной организации |  |
| 1.3 | Провайдер ИНН | 1655258235 |
| 1.4 | Ответственный за программу ФИО | Бадрутдинова Гульгена Эдуардовна |
| 1.5 | Ответственный должность | Менеджер проектов АНО ВО "Университета Иннополис |
| 1.6 | Ответственный Телефон | 89953315878 |
| 1.7 | Ответственный Е-mail | gulgenabad@gmail.com |

1. **Основные Данные**

|  |  |  |
| --- | --- | --- |
| **№** | **Название** | **Описание** |
| 2.1 | Название программы | Разработчик JAVA. Продолжающий уровень |
| 2.2 | Ссылка на страницу программы | https://java.innopolis.university/ |
| 2.3 | Формат обучения | Онлайн |
|  | Подтверждение от ОО наличия возможности реализации образовательной программы с применением электронного обучения и (или) дистанционных образовательных технологий с возможностью передачи данных в форме элементов цифрового следа | Программа реализуется с применением электронного обучения и дистанционных образовательных технологий. Лекционные и практические занятия проводятся в форме вебинаров. Задания и материалы программы размещаются в системе LMS Moodle, доступ к которой получают все обучающиеся. |
| 2.4 | Уровень сложности | Продолжающий |
| 2.5 | Количество академических часов | **72** |
|  | Практикоориентированный характер образовательной программы: не менее 50 % трудоёмкости учебной деятельности отведено практическим занятиям и (или) выполнению практических заданий в режиме самостоятельной работы (кол-во академических часов) | |  | | --- | | 50 академических часов. Из них на выполнение практических заданий - 25 часов, на выполнение самостоятельных работ-25 часов | |
| 2.6 | Стоимость обучения одного обучающегося по образовательной программе, а также предоставление ссылок на 3 (три) аналогичные образовательные программы иных организаций, осуществляющих обучение, для оценки объективности стоимости или обоснование уникальности представленной образовательной программы в случае отсутствия аналогичных образовательных программ на рынке образовательных услуг | **Стоимость обучения:** 30 000 рублей 00 коп. за одного обучающегося по ОО  **1.** [https://otus.ru](https://otus.ru/)  **2.** [https://course.skillbox.ru](https://go.acstat.com/e2540d514cac6a94)  **3.** [https://www.specialist.ru](https://www.specialist.ru/)  **Обоснование:**  Курс «Разработчик Java. Продолжающий уровень» позволяет самостоятельно заниматься разработкой небольших проектов, включая в работу junior разработчиков, создавать более эффективный и простой код, избегая простых ошибок/багов. и обзору современных тенденций в программировании.  Учебный курс включает в себя 12 онлайн-занятий (вебинары) с преподавателем.  Что будет на курсе:   * Условные операторы. * Циклы. * Одномерные массивы. * Простые сортировки. * Сортировка подсчетом. * Двумерные массивы. * Простые структуры данных. Списки, стеки, очереди. * Работа со строками. * Сложные сортировки.   Закрепление теоретического материала осуществляется через выполнение практических заданий с автоматизированной проверкой, тренажеров. К каждому занятию прилагается полный конспект, который загружается в платформу, что позволяет сделать изучение курса комфортной. |
| 2.7 | Минимальное количество человек на курсе | 100 |
| 2.8 | Максимальное количество человек на курсе | 400 |
| 2.9 | Данные о количестве слушателей, ранее успешно прошедших обучение по образовательной программе | При наличии |
| 2.10 | Формы аттестации | Текущая аттестация - самоконтроль;  Итоговая аттестация - тестирование |
|  | Указание на область реализации компетенций цифровой экономики, к которой в большей степени относится образовательная программа, в соответствии с Перечнем областей | Программирование и создание ИТ-продуктов  Разработка компьютерных игр и мультимедийных приложений  Разработка мобильных приложений |

1. **Аннотация программы**

Курс «Разработчик Java. Продолжающий уровень» является продолжением курса «Разработчик Java. Начинающий уровень», «Основы программирования. Быстрый старт» и других программ начального уровня, знакомящих с принципами программирования. Освоение программы позволяет освоить начальный уровень алгоритмов и структур данных, позволяет закрепить навыки кодирования и проектирования программного кода. Полученные знания и навыки находят постоянное применение в разных сферах, связанных с IT, а именно: мобильная разработка, гейм разработка, разработка ПО и прочие.

Требования и рекомендации для обучения по образовательной программе;

Прохождение курса «Разработчик Java. Начальный уровень» или любых других курсов и программ начального уровня;

Выполнение не менее 50% заданий входного тестирования.

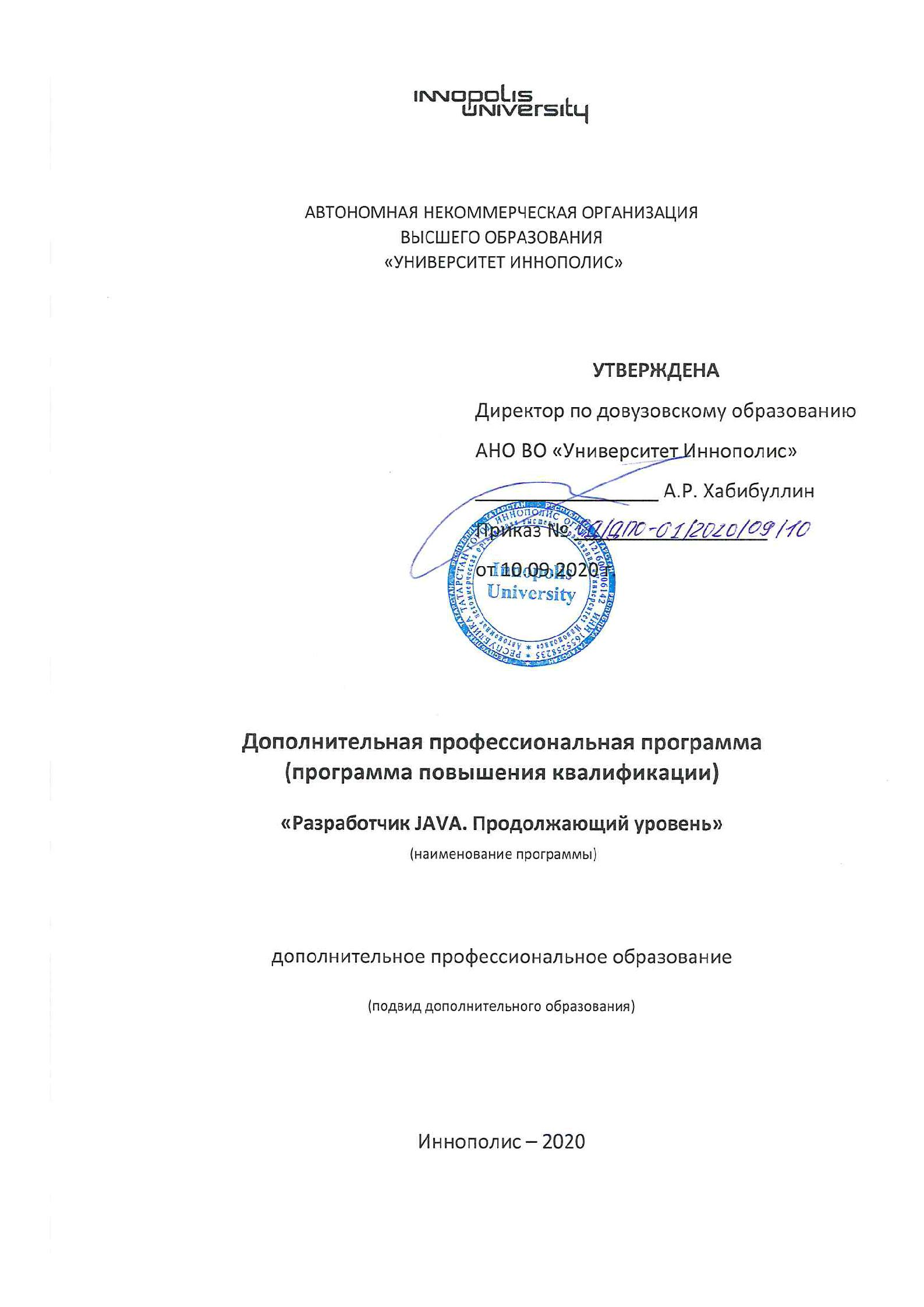
**Результатами обучения на курсе являются:**

В результате освоения образовательной программы слушатель будет знать такие понятия как цикл (while, for in), одномерный и двумерный массив, сортировка, стек, очередь, список; основные концепции программирования сложных алгоритмов; методы работы со строками; типы сложных сортировок.

Слушатель научится решать алгоритмические задачи с использованием одномерных и двумерных массивов, сортировок на языке программирования Java; использовать списки ; формально выполнять алгоритмы, описанные с использованием простых и сложны сортировок, одномерных и двумерных массивов, циклов; использовать готовые прикладные компьютерные программы и сервисы; создавать и выполнять программы для решения сложных алгоритмических задач в программе Java.

Получит навыки работы с персональным компьютером; самостоятельного построения компьютерной модели; беглого компьютерного ввода на русском и английском языках; свободного владения компьютерным графическим интерфейсом; применения сложных алгоритмов при решении задач на языке Java; использования теории алгоритмов при решении олимпиадных задач; выбора способа представления данных в зависимости от постановленной задачи.





**ОБЩАЯ ХАРАКТЕРИСТИКА ПРОГРАММЫ**

**1.Цель программы**

Целью подготовки слушателей по Программе является повышение квалификации обучающихся за счет создания условий для их саморазвития через решение алгоритмических задач с использованием одномерных и двумерных массивов, строк, сортировок.

**2.Планируемые результаты обучения:**

2.1. Знание (осведомленность в областях)

2.1.1. Понятие цикл (while, for in), одномерный и двумерный массив, сортировка, стек, очередь, список;

2.1.2. Основные концепции программирования сложных алгоритмов;

2.1.3. Методы работы со строками;

2.1.4. Типы сложных сортировок.

2.2. Умение (способность к деятельности)

2.2.1. Решать сложные алгоритмические задачи с использованием одномерных и двумерных массивов, сортировок на языке программирования Java;

2.2.2. Использовать списки;

2.2.3. Формально выполнять алгоритмы, описанные с использованием простых и сложны сортировок, одномерных и двумерных массивов, циклов;

2.2.4. Использовать готовые прикладные компьютерные программы и сервисы;

2.2.5. Создавать и выполнять программы для решения сложных алгоритмических задач в программе Java;

2.3. Навыки (использование конкретных инструментов)

2.3.1 Работы с персональным компьютером;

2.3.2. Самостоятельного построения компьютерной модели;

2.3.3. Беглого компьютерного ввода на русском и английском языках;

2.3.4. Свободного владения компьютерным графическим интерфейсом;

2.3.5. Использования теории алгоритмов при решении задач;

2.3.6. Выбора способа представления данных в зависимости от постановленной задачи.

**3.Категория слушателей**

3.1 Образование: высшее или среднее профессиональное

3.2. Наличие опыта профессиональной деятельности: не требуется

**4.Учебный план программы «Разработчик JAVA. Продолжающий уровень»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Модуль** | **Всего, час** | **Виды учебных занятий** | | |
| **Лекции** | **Практические занятия** | **Самостоятельная работа** | |
| 1. | **Условные операторы** | **4** | **1** | **2** | **1** | |
| 1.1. | Простые, операторы сравнения; условные операторы if, if/else  Решение задач с использованием условных операторов и операторов сравнения | 4 | 1 | 2 | 1 | |
| 2. | **Циклы** | **4** | **1** | **2** | **1** | |
| 2.1. | Циклы while, for in;  Бесконечные циклы; вложенные циклы | 2 | 1 | 1 |  | |
| 2.2. | Решение задач на составление простых алгоритмов | 2 |  | 1 | 1 | |
| 3. | **Одномерные массивы** | **8** | **1** | **5** | **2** | |
| 3.1. | *Основные операторы программирования.*  *Понятие структуры данных – массив.* | 4 | **1** | **2** | **1** | |
| 3.2. | *Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике.* | 4 |  | **3** | **1** | |
| 4. | **Простые сортировки.** | **6** | **1** | **4** | **1** | |
| 4.1. | Сортировка пузырьком; сортировка вставками | 2 | 1 | 1 |  | |
| 4.2. | Способы реализации сортировок; время сортировок; функция сортировки пузырьком | 1 |  | 1 |  | |
| 4.3. | Решение задач с использованием сортировок пузырьком и вставками. | 3 |  | 2 | 1 | |
| 5. | **Сортировка подсчетом** | **8** | **1** | **4** | **3** | |
| 5.1. | *Сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки.* | 4 | 1 | 2 | 1 | |
| 5.2. | Решение задач с использованием сортировки подсчетом. | 4 |  | 2 | 2 | |
| 6. | **Двумерные массивы.** | **8** | **1** | **6** | **1** | |
| 6.1. | Понятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы. | 4 | 1 | 3 |  | |
| 6.2. | Решение задач на двумерные массивы и матрицы. | 4 |  | 3 | 1 | |
| 7. | **Простые структуры данных. Списки, стеки, очереди.** | **8** |  | **6** | **2** | |
| 7.1. | Формирование строк; строковые функции и методы, операции над строками; обработка строк  Использование строк при решении задач и создании проектов | 8 |  | 6 | 2 | |
| 8. | **Работа со строками.** | **8** |  | **6** | **2** | |
| 8.1. | Формирование строк; строковые функции и методы, операции над строками; обработка строк | 4 |  | 3 | 1 | |
| 8.2. | Использование строк при решении задач и создании проектов | 4 |  | 3 | 1 | |
| 9. | **Сложные сортировки.** | **8** |  | **6** | **2** | |
| 9.1. | Быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок. | 4 |  | 3 | 1 | |
| 9.2. | Использование сортировок при решении задач и создании проектов | 4 |  | 3 | 1 | |
| 10. | **Индивидуальные проекты.** | **8** |  | **6** | **2** | |
| 10.1. | *Закрепление пройденного материала на примере создания собственного проекта.* | 2 |  | **2** |  | |
| 10.2. | *Обучение навыкам создания проектов и работа с большим участком кода.* | 6 |  | **4** | **2** | |
| 11. | **Итоговая аттестация** | **2** |  |  |  | |

**5. Календарный план-график реализации образовательной «Разработчик JAVA. Продолжающий уровень»**

с 01.11.2020 – 13.11.2020 в текущем календарном году, указания на периодичность набора групп (не менее 1 группы в месяц)

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование учебных модулей** | **Трудоёмкость (час)** | **Сроки обучения** |
| **1** | Условные операторы | 4 | 1.11.2020 |
| **2** | Циклы | 4 | 2.11.2020 |
| **3** | Одномерные массивы | 8 | 3.11-4.11.2020 |
| **4** | Простые сортировки | 6 | 5.11.2020 |
| **5** | Сортировка подсчетом | 8 | 6.11-7.11.2020 |
| **6** | Двумерные массивы | 8 | 7.11.2020 |
| **7** | Простые структуры данных. Списки, стеки, очереди | 8 | 8.11-9.11.2020 |
| **8** | Работа со строками | 8 | 10.11.-11.11.2020 |
| **9** | Сложные сортировки. | 8 | 11.11.2020 |
| **10** | Индивидуальные проекты | 8 | 12.11.2020 |
| **11** | Итоговая аттестация | 2 | 13.11.2020 |
| **Всего:** | | **72** |  |

**6.Учебно-тематический план программы «Разработчик JAVA. Продолжающий уровень»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Модуль / Тема** | **Всего, час** | **Виды учебных занятий** | | | **Формы контроля** |
| **лекции** | **практические занятия** | **самостоятельная работа** |
| 1. | **Условные операторы** | **4** | **1** | **2** | **1** | **Текущий контроль** |
| 1.1. | Простые, операторы сравнения; условные операторы if, if/else  Решение задач с использованием условных операторов и операторов сравнения | 4 | 1 | 2 | 1 |  |
| 2. | **Циклы** | **4** | **1** | **2** | **1** | **Текущий контроль** |
| 2.1. | Циклы while, for in;  Бесконечные циклы; вложенные циклы | 2 | 1 | 1 |  |  |
| 2.2. | Решение задач на составление простых алгоритмов | 2 |  | 1 | 1 |  |
| 3. | **Одномерные массивы** | **8** | **1** | **5** | **2** | **Текущий контроль** |
| 3.1. | *Основные операторы программирования.*  *Понятие структуры данных – массив.* | 4 | **1** | **2** | **1** |  |
| 3.2. | *Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике.* | 4 |  | **3** | **1** |  |
| 4. | **Простые сортировки.** | **6** | **1** | **4** | **1** | **Текущий контроль** |
| 4.1. | Сортировка пузырьком; сортировка вставками | 2 | 1 | 1 |  |  |
| 4.2. | Способы реализации сортировок; время сортировок; функция сортировки пузырьком | 1 |  | 1 |  |  |
| 4.3. | Решение задач с использованием сортировок пузырьком и вставками. | 3 |  | 2 | 1 |  |
| 5. | **Сортировка подсчетом** | **8** | **1** | **4** | **3** | **Текущий контроль** |
| 5.1. | *Сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки.* | 4 | 1 | 2 | 1 |  |
| 5.2. | Решение задач с использованием сортировки подсчетом. | 4 |  | 2 | 2 |  |
| 6. | **Двумерные массивы.** | **8** | **1** | **6** | **1** | **Текущий контроль** |
| 6.1. | Понятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы. | 4 | 1 | 3 |  |  |
| 6.2. | Решение задач на двумерные массивы и матрицы. | 4 |  | 3 | 1 |  |
| 7. | **Простые структуры данных. Списки, стеки, очереди.** | **8** |  | **6** | **2** | **Текущий контроль** |
| 7.1. | Формирование строк; строковые функции и методы, операции над строками; обработка строк  Использование строк при решении задач и создании проектов | 8 |  | 6 | 2 |  |
| 8. | **Работа со строками.** | **8** |  | **6** | **2** | **Текущий контроль** |
| 8.1. | Формирование строк; строковые функции и методы, операции над строками; обработка строк | 4 |  | 3 | 1 |  |
| 8.2. | Использование строк при решении задач и создании проектов | 4 |  | 3 | 1 |  |
| 9. | **Сложные сортировки.** | **8** |  | **6** | **2** | **Текущий контроль** |
| 9.1. | Быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок. | 4 |  | 3 | 1 |  |
| 9.2. | Использование сортировок при решении задач и создании проектов | 4 |  | 3 | 1 |  |
| 10. | **Индивидуальные проекты.** | **8** |  | **6** | **2** | **Текущий контроль** |
| 10.1. | *Закрепление пройденного материала на примере создания собственного проекта.* | 2 |  | **2** |  |  |
| 10.2. | *Обучение навыкам создания проектов и работа с большим участком кода.* | 6 |  | **4** | **2** |  |

**7. Учебная (рабочая) программа повышения квалификации «Разработчик JAVA. Продолжающий уровень»**

**Условные операторы.**

Краткое содержание лекции: Простые, операторы сравнения; условные операторы if, if/else

Практические занятия: решение задач с использованием условных операторов и операторов сравнения

**Циклы.**

Краткое содержание лекции: Циклы while, for in;

Практические занятия направлены на освоение бесконечных циклов; вложенных циклов; решение задач на составление простых алгоритмов.

**Одномерные массивы.**

Краткое содержание лекции: Основные операторы программирования.

Практические занятия: Понятие структуры данных – массив.

Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике.

**Простые сортировки.**

Краткое содержание лекции: сортировка пузырьком; сортировка вставками; способы реализации сортировок; время сортировок; функция сортировки пузырьком

Практические занятия направлены на решение задач с использованием сортировок пузырьком и вставками.

**Сортировка подсчетом.**

Краткое содержание лекции: сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки.

Практические занятия направлены на решение задач с использованием сортировки подсчетом.

**Двумерные массивы.**

Краткое содержание лекции: понятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы.

Практические занятия подразумевают решение задач на двумерные массивы и матрицы.

**Простые структуры данных. Списки, стеки, очереди.**

Краткое содержание лекции: формирование строк; строковые функции и методы, операции над строками; обработка строк

Практические занятия направлены на использование строк при решении задач и создании проектов

**Работа со строками.**

Краткое содержание лекции: формирование строк; строковые функции и методы, операции над строками; обработка строк

Практические занятия направлены на использование строк при решении задач и создании проектов

**Сложные сортировки.**

Краткое содержание лекции: быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок.

На практических занятиях рассматривается использование сортировок при решении задач и создании проектов

**Индивидуальные проекты.**

Практические занятия: Закрепление пройденного материала на примере создания собственного проекта. Обучение навыкам создания проектов и работа с большим участком кода.

**Описание практико-ориентированных заданий и кейсов**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Номер темы/модуля** | **Наименование практического занятия** | **Описание** |
|  | **Условные операторы.** | Решение задач с использованием условных операторов и операторов сравнения | Работа в среде разработки Intelij IDEA. Решение задач с использованием условных операторов и операторов сравнения. |
|  | **Циклы.** | Освоение бесконечных циклов; вложенных циклов; решение задач на составление простых алгоритмов. | Работа в среде разработки Intelij IDEA. Решение задач с использованием циклов. |
|  | **Одномерные массивы.** | Понятие структуры данных – массив.  Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике. | Работа в среде разработки Intelij IDEA. Решение задач с использованием одномерных массивов. Создание собственной простейшей компьютерной графики. |
|  | **Простые сортировки.** | Рзадач с использованием сортировок пузырьком и вставками. | Работа в среде разработки Intelij IDEA. Решение задач с использованием сортировок пузырьком и вставками. |
|  | **Сортировка подсчетом.** | Решение задач с использованием сортировки подсчетом. | Работа в среде разработки Intelij IDEA. Решение задач с использованием сортировки подсчетом. |
|  | **Двумерные массивы.** | Решение задач на двумерные массивы и матрицы | Работа в среде разработки Intelij IDEA. Решение задач с использованием двумерных массивов и матриц. |
|  | **Простые структуры данных. Списки, стеки, очереди.** | Использование списков, стеков и очередей при решении задач и создании проектов | Работа в среде разработки Intelij IDEA. Решение задач с использованием списков, стеков и очередей. Создание простого проекта. |
|  | **Работа со строками.** | Использование строк при решении задач и создании проектов | Работа в среде разработки Intelij IDEA. Решение задач с использованием строк. |
|  | **Сложные сортировки.** | Использование сортировок при решении задач и создании проектов | Работа в среде разработки Intelij IDEA. Решение задач с использованием сложных сортировок. Создание простого проекта. |
|  | **Индивидуальные проекты.** | Закрепление пройденного материала на примере создания собственного проекта. Обучение навыкам создания проектов и работа с большим участком кода. | Работа в среде разработки Intelij IDEA. Создание собственных проектов на основе пройденного материала. |

**8.Оценочные материалы по образовательной программе**

**8.1. Вопросы тестирования по модулям**

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование раздела (модуля)** | **Задание** | **Критерии оценки** |
|  | **Входной тест;** | ***Задача 1***. Упорядочить 3 числа.  Дано три числа, записанный в отдельных строках. Упорядочите их в порядке неубывания. Программа должна считывать три числа a, b, c, затем программа должна менять их значения так, чтобы стали выполнены условия a <= b <= c, затем программа выводит тройку a, b, c.  Входные данные  Вводятся три числа, каждое записано в отдельной строке.  Выходные данные  Выведите ответ на задачу.  Примеры  входные данные  1  2  1  выходные данные  1 1 2  ***Задача*** 2. Четные числа.  Входные данные  Вводятся целые числа a и b. Гарантируется, что a не превосходит b  Выходные данные  Выведите (через пробел) все четные числа от a до b (включительно).  Примеры  входные данные  2  5  выходные данные  2 4  ***Задача 3.*** Остаток.  Входные данные  Вводятся 4 числа: a, b, c и d.  Выходные данные  Выведите все числа на отрезке от a до b, дающие остаток c при делении на d. Если таких чисел не существует, то ничего выводить не нужно.  Примеры  входные данные  2  5  0  2  выходные данные  2 4  ***Задача 4.*** Делители числа.  Выведите все натуральные делители числа x в порядке возрастания (включая 1 и само число).  Входные данные  Вводится натуральное число x  Выходные данные  Выведите все делители числа x  Примеры  входные данные  32  выходные данные  1 2 4 8 16 32  ***Задача 5.*** Нули.  Входные данные  Вводится число N, а затем N чисел.  Выходные данные  Подсчитайте и выведите, сколько среди данных N чисел нулей.  Примеры  входные данные  3  1  2  3  выходные данные  0  ***Задача 6***. Подсчет чисел.  Подсчитайте, сколько среди данных N чисел нулей, положительных чисел, отрицательных чисел.  Входные данные  Вводится число N, а затем N целых чисел.  Выходные данные  Необходимо вывести сначала число нулей, затем число положительных и отрицательных чисел.  Примеры  входные данные  5  28  0  0  0  0  выходные данные  4 1 0 | Слушатель способен решить более 60% задач входного теста |
| 1. | Условные операторы. | Даны координаты двух точек на плоскости, требуется определить, лежат ли они в одной координатной четверти или нет (все координаты отличны от нуля).  Входные данные  Вводятся 4 числа: координаты первой точки (x1, y1) и координаты второй точки (x2, y2).  Выходные данные  Программа должна вывести слово YES, если точки находятся в одной координатной четверти, в противном случае вывести слово NO. | Слушатель усвоил решение по данной теме с использованием условных операторов. |
| 2. | Циклы. | Входные данные  Вводятся 4 числа: a, b, c и d.  Выходные данные  Выведите все числа на отрезке от a до b, дающие остаток c при делении на d. Если таких чисел не существует, то ничего выводить не нужно. | Слушатель усвоил способы решения задач по данной теме с использованием циклов. |
| 3. | Одномерные массивы. | На клетчатой бумаге Петя нарисовал отрезок из точки с координатами (a,b) в точку с координатами (c,d). Через сколько клеток проходит этот отрезок (считается, что отрезок проходит через клетку, если он проходит через ее внутренность, если же он проходит только через вершину или по границе клетки, считается, что он не проходит через клетку).  Входные данные  Вводятся целые числа a, b, c, d. Числа по модулю не превышают 109.  Выходные данные  Выведите одно число — количество клеток, через которые проходит отрезок. | Слушателем усвоены способы решения задач по данной теме с использованием одномерных массивов. |
| 4. | Простые сортировки. | Задана матрица K, содержащая n строк и m столбцов. Седловой точкой этой матрицы назовем элемент, который одновременно является минимумом в своей строке и максимумом в своем столбце.  Найдите количество седловых точек заданной матрицы. | Слушатель усвоил способы решения задач по данной теме с использованием различных сортировок. |
| 5. | Сортировка подсчетом. | Реализуйте алгоритм сортировки подсчетом для произвольных чисел, по модулю не превосходящих 10000.  Входные данные  На вход программе сначала подается значение n ≤ 100000 – количество элементов в массиве. В следующей строке входных данных распо-ложены сами элементы массива – целые числа, по модулю не превосходящие 10000.  Выходные данные  Распечатайте отсортированный по неубыванию массив. | Слушатель усвоил решение задач по данной теме с сортировки подсчетом. |
| 6. | Двумерные массивы. | Дано число n, n 100. Создайте массив n×n и заполните его по следующему правилу:  - числа на диагонали, идущей из правого верхнего в левый нижний угол, равны 1;  - числа, стоящие выше этой диагонали, равны 0;  - числа, стоящие ниже этой диагонали, равны 2.  Входные данные  Программа получает на вход число n.  Выходные данные  Необходимо вывести полученный массив. Числа разделяйте одним пробелом. | Слушатель усвоил решение задач по данной теме, используя двумерные массивы. |
| 7. | Простые структуры данных. Списки, стеки, очереди. | Реализуйте структуру данных "стек". Напишите программу, содержащую описание стека и моделирующую работу стека, реализовав все указанные здесь методы. Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку. | Слушатель усвоил решение задач по данной теме с использованием простых структур данных. |
| 8. | Работа со строками. | Дана строка, состоящая ровно из двух слов, разделенных пробелом. Переставьте эти слова местами. Результат запишите в строку и выведите получившуюся строку.  При решении этой задачи нельзя пользоваться циклами и инструкцией if.  Входные данные  Вводится строка.  Выходные данные  Выведите ответ на задачу.  Примеры  входные данные  Hello, world!  выходные данные  world! Hello, | Слушатель усвоил решение задач такого типа с использованием строк. |
| 9. | Сложные сортировки. | Переставьте элементы данного списка в обратном порядке, затем выведите элементы полученного списка.  Эта задача отличается от предыдущей тем, что вам нужно изменить значения элементов самого списка, поменяв местами A[0] c A[n-1], A[1] с A[n-2], а затем вывести элементы списка подряд.  Входные данные  Вводится список чисел. Все числа списка находятся на одной строке.  Выходные данные  Выведите ответ на задачу.  Примеры  входные данные  1 2 3 4 5  выходные данные  5 4 3 2 1 | Слушатель усвоил решение задач по данной теме с использованием сложных сортировок. |
| 10. | Индивидуальные проекты. | Разработать индивидуальный проект в соответствии с выбранной темой. | Слушатель должен разработать собственный проект |
| 11. | Итоговая аттестация  Контрольный тест | Написать проект, который представляет собой программу, позволяющую стабильно работать в сети Интернет. Примерная тематика:  1. Развлекательная программа (игра, анимация, клип и т. п.)  2. Мобильное приложение  **Итоговый контрольный тест:**  ***Задача 1****.* Дано число N. Определите, сколькими способами можно расставить на доске N×N N ферзей, не бьющих друг друга.  Входные данные  Задано единственное число N. (N ≤ 10)  Выходные данные  Необходимо вывести количество способов, которыми можно расставить на доске N×N N ферзей, не бьющих друг друга.  Примеры  входные данные  8  выходные данные  92  ***Задача 2.*** На прямой задано некоторое множество отрезков с целочисленными координатами концов [Li, Ri]. Выберите среди данного множества подмножество отрезков, целиком покрывающее отрезок [0, M], (M — натуральное число), содержащее наименьшее число отрезков.  Входные данные  В первой строке указана константа M (1M5000). В каждой последующей строке записана пара чисел Li и Ri (LiRi50000), задающая координаты левого и правого концов отрезков. Список завершается парой нулей. Общее число отрезков не превышает 100 000.  Выходные данные  В первой строке выходного файла выведите минимальное число отрезков, необходимое для покрытия отрезка [0; M]. Далее выведите список покрывающего подмножества, упорядоченный по возрастанию координат левых концов отрезков. Список отрезков выводится в том же формате, что и во входe. Завершающие два нуля выводить не нужно. Если покрытие отрезка [0, M] исходным множеством отрезков [Li, Ri] невозможно, то следует вывести единственную фразу “No solution”.  Примеры  входные данные  1  -1 0  -5 -3  2 5  0 0  выходные данные  No solution  входные данные  1  -1 0  0 1  0 0  выходные данные  1  0 1  ***Задача 3.*** Дан массив. Надо научиться обрабатывать два типа запросов.  \* 1 L R - перевернуть отрезок [LR]  \* 2 L R - найти минимум на отрезке [LR]  Входные данные  Первая строка файла содержит два числа n, m. (1nm105) Во второй строке находится n чисел ai (1ai109) — исходный массив. Остальные m строк содержат запросы, в формате описанном в условии. Для чисел L, R выполняется ограничение (1LRn).  Выходные данные  На каждый запрос типа 2, во входной файл выведите ответ на него, в отдельной строке.  Примеры  входные данные  10 7  5 3 2 3 12 6 7 5 10 12  2 4 9  1 4 6  2 1 8  1 1 8  1 8 9  2 1 7  2 3 6  выходные данные  3  2  2  2  ***Задача 4***. Дан граф из n вершин, раскрасьте его в минимально возможное число цветов так, чтобы никакие две вершины, соединенные ребром, не были одного цвета.  Входные данные  В первой строке содержится число t - количество тестовых примеров (1t5).  Далее содержится t тестовых случаев, заданных в следующем формате:  В первой строке записаны числа n и m - количество вершин и ребер соответственно (1n17, 0m2n(n−1)).  Затем идет m строк, в которых содержится по два числа vi ui, что означает, что вершины vi и ui соеденены ребром (1viuinvi=ui).  Гарантируется, что все ребра в каждом тестовом случае различны.  Выходные данные  Для каждого тестового случая в первой строке выведите минимальное число цветов k.  Во второй строке выведите n чисел ai - цвета вершин (1aik).  Примеры  входные данные  3  3 3  1 2  2 3  3 1  5 3  2 1  3 1  4 2  6 7  1 2  1 5  2 5  2 3  2 4  5 6  5 4  выходные данные  3  3 2 1  2  1 2 2 1 1  3  1 3 1 1 2 1  ***Задача 5.*** Представьте данное число n в виде суммы двух кубов.  Входные данные  Программа получает на вход одно натуральное число n(n <= 1028).  Выходные данные  Программа должна вывести 2 целых неотрицательных числа, сумма кубов которых равна n. Если это невозможно, выведите строку impossible.  Примеры  входные данные  2  выходные данные  1 1  входные данные  3  выходные данные  impossible  ***Задача 6.*** Задана матрица K, содержащая n строк и m столбцов. Седловой точкой этой матрицы назовем элемент, который одновременно является минимумом в своей строке и максимумом в своем столбце.  Найдите количество седловых точек заданной матрицы.  Входные данные  Первая строка содержит целые числа n и m (1 ≤ n, m ≤ 750). Далее следуют n строк по m чисел в каждой. j-ое число i-ой строки равно kij. Все kij по модулю не превосходят 1000.  Выходные данные  Выведите ответ на задачу.  Примеры  входные данные  2 2  0 0  0 0  выходные данные  4  входные данные  2 2  1 2  3 4  выходные данные  1 | Защита итоговой проектной работы, написанной на языке Java, стабильно работающей в сети Интернет.  Слушатель успешно решил 60% контрольного теста: |

**9.Организационно-педагогические условия реализации программы**

**9.1. Кадровое обеспечение программы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№**  **п/п** | **Фамилия, имя, отчество (при наличии)** | **Место основной работы и должность, ученая степень и ученое звание (при наличии)** | **Ссылки на веб-страницы с портфолио (при наличии)** | **Фото в формате jpeg** | **Отметка о полученном согласии на обработку персональных данных** |
| 1 | Бусыгина Ксения Александровна | Университет Иннополис, специалист по учебно-методической работе по информатике | - |  | согласна |

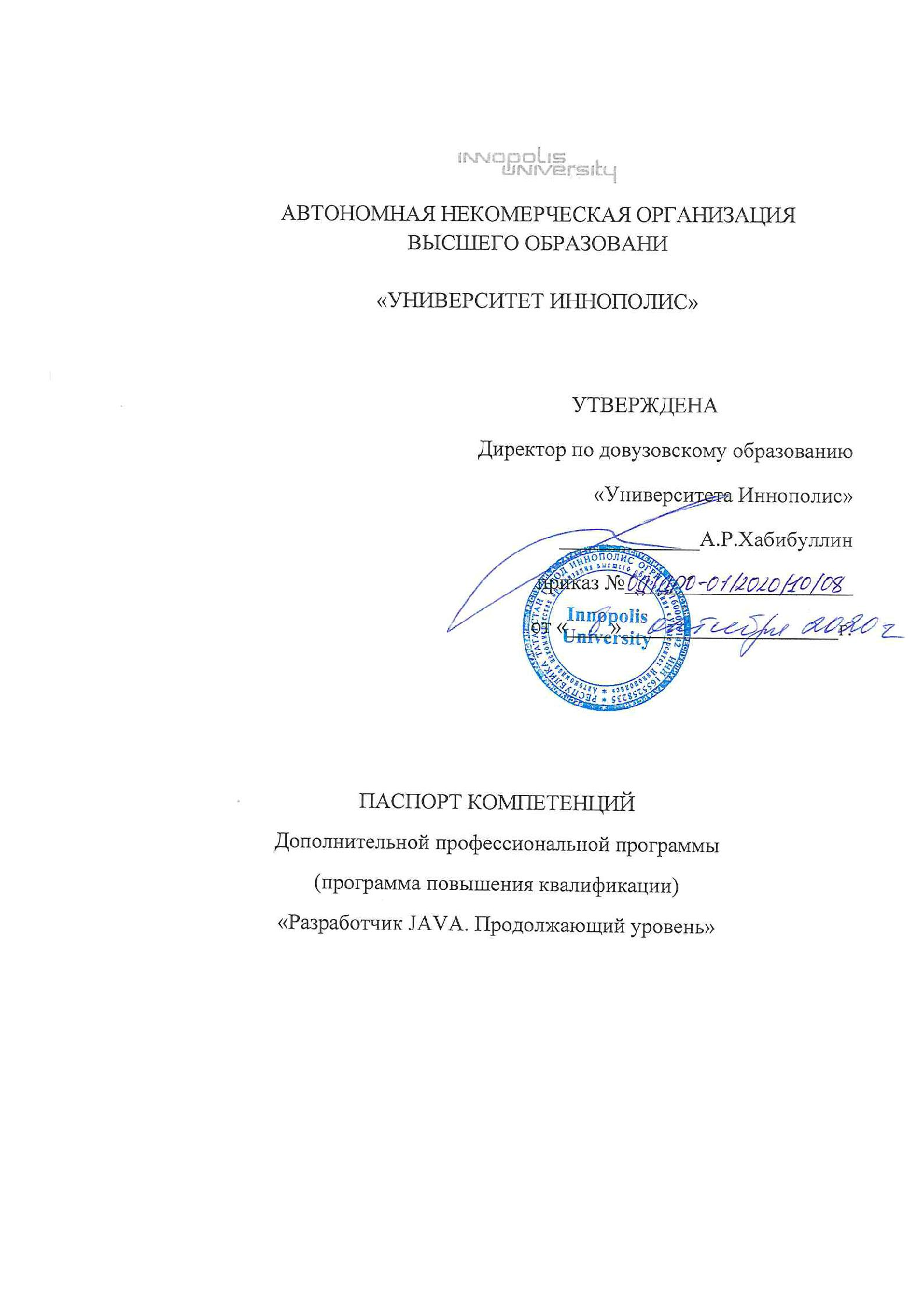
**9.2. Учебно-методическое обеспечение и информационное сопровождение**

|  |  |
| --- | --- |
| **Учебно-методические материалы** | |
| Методы, формы и технологии | Методические разработки,  материалы курса, учебная литература |
| Технологии дифферецированного обучения, предметно-ориентировочные, личностно-ориентированные, разноуровневого обучения, информационно-коммуникативные технологии | Материалы курса  https://innoclass.innopolis.university/ru/profile/course/72/  Учебная литература  1. Сузи, Р. А. Язык программирования Python / Р. А. Сузи. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 350 c. — ISBN 5-9556-0058-2. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: http://www.iprbookshop.ru/52211.html  2. Томашевский, П. Р. Привет, Python! Моя первая книга по программированию / П. Р. Томашевский. — СПб.: Наука и Техника, 2018. — 256 c. — ISBN 978-5-94387-748-3. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: http://www.iprbookshop.ru/73047.html  3. Буйначев, С. К. Основы программирования на языке Python: учебное пособие / С. К. Буйначев, Н. Ю. Боклаг; под редакцией Ю. В. Песин. — Екатеринбург: Уральский федеральный университет, ЭБС АСВ, 2014. — 92 c. — ISBN 978-5-7996-1198-9. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: http://www.iprbookshop.ru/66183.html  4. HTML, CSS, SCRATCH, PYTHON. Моя первая книга по программированию / С. В. Голиков, Е. В. Дубовик, П. С. Русин, Ю. А. Иркова. — СПб.: Наука и Техника, 2018. — 336 c. — ISBN 978-5-94387-754-4. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: http://www.iprbookshop.ru/78106.html |

|  |  |
| --- | --- |
| **Информационное сопровождение** | |
| Электронные  образовательные ресурсы | Электронные  информационные ресурсы |
| Гаврилов, А.В. Программирование на Java. Конспект лекций [Электронный ресурс]: учебное пособие / А. В. Гаврилов, С. В. Клименков, Е.А. Цопа. — Электрон. дан. — Санкт-Петербург: НИУ ИТМО, 2010. — 130 с. — Режим доступа: https://e.lanbook.com/book/43547 | Образовательная платформа InnoClass  https://innoclass.innopolis.university/ |
| Мейер Б. Инструменты, алгоритмы и структуры данных. М.: ИНТУИТ, 2016. – 542 с. [Электронный ресурс] http://www.iprbookshop.ru/73680.html | Образовательный портал Информатикс: дистанционная подготовка к олимпиадам по информатике  https://informatics.mccme.ru/ |
| Фарафонов, А. С. Программирование на языке высокого уровня: методические указания к проведению лабораторных работ по курсу «Программирование» / А. С. Фарафонов. — Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013. — 32 c. — ISBN 2227-8397. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: http://www.iprbookshop.ru/22912.html | Система электронного обучения и тестирования MOODLE  https://stem.university.innopolis.ru/ |
| Дубаков, А.А. Введение в объектно-ориентированное программирование на Java [Электронный ресурс]: учебное пособие / А.А. Дубаков. — Электрон. дан. — Санкт-Петербург: НИУ ИТМО, 2016. — 250 с. — Режим доступа: https://e.lanbook.com/book/110468 | Система электронного тестирования psmc  https://pcms.university.innopolis.ru/ |

**9.3. Материально-технические условия реализации программы**

|  |  |
| --- | --- |
| **Вид занятий** | **Наименование оборудования,**  **программного обеспечения** |
| Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом;  Специальные помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории;  Для проведения занятий лекционного и семинарского (практического) типа предлагаются наборы демонстрационного оборудования и учебно-наглядных пособий, обеспечивающие тематические иллюстрации, соответствующие программе дисциплины (модуля).  Лаборатории Университета оснащены лабораторным и специализированным оборудованием. | |
| *Лекция* | *ПК, имеющий доступ в сеть Интернет;* среда разработки Intelij IDEA*; средство удаленной связи Zoom* |
| *Практическая работа* | *ПК, имеющий доступ в сеть Интернет;* среда разработки Intelij IDEA*; средство удаленной связи Zoom* |
| *Самостоятельная работа* | *ПК, имеющий доступ в сеть Интернет;* среда разработки Intelij IDEA*; средство удаленной связи Zoom* |
| *Итоговая аттестация* | *ПК, имеющий доступ в сеть Интернет;* среда разработки Intelij IDEA*; средство удаленной связи Zoom* |



|  |  |
| --- | --- |
|  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Наименование компетенции | | Способность писать оптимальный программный код в соответствии с техническим заданием и оценивать сроки выполнения. | |
| 2. | Указание типа компетенции | ПК -1 профессиональный | Производственно-технологический | |
| 3 | Определение, содержание, основные сущностные характеристики компетенции | | Под компетенцией ПК-1 понимается способность написания программного кода с использованием языков программирования, определения и манипулирования данными.  Структурными элементами данной компетенции являются:   * создание программного кода в соответствии с техническим заданием (готовыми спецификациями) * оптимизация программного кода с использованием специализированных программных средств * оценка и согласование сроков выполнения поставленных задач | |
| 4 | Дескриптор знаний, умений и навыков по уровням | | Уровни сформированности компетенции обучающегося | Индикаторы |
|  | **Уметь:**  применять выбранные языки программирования для написания программного кода,  использовать выбранную среду программирования и средства системы, управления базами данных. Использовать возможности имеющейся технической и/или программной архитектуры  **Знать:**  синтаксис выбранного языка программирования, особенности программирования на этом языке, стандартные библиотеки языка программирования, методологии разработки программного обеспечения, методологии и технологии проектирования и использования баз данных, технологии программирования, особенности выбранной среды программирования и системы управления базами данных, компоненты программно-технических архитектур, существующие приложения и интерфейсы взаимодействия с ними  **Необходимые навыки:**  владеть стандартными средствами базовых информационных  процессов и технологий;  владеть моделями и средствами разработки архитектуры  информационных систем. | | Базовый | Имеет навыки работы с персональным компьютером;  Имеет навыки самостоятельного построения компьютерной модели;  Имеет навыки беглого компьютерного ввода на русском и английском языках;  Имеет навыки свободного владения компьютерным графическим интерфейсом;  Имеет навыки применения алгоритмов при решении задач на языке Java; использования теории алгоритмов при решении олимпиадных задач;  Имеет навыки выбора способа представления данных в зависимости от постановленной задачи. |
|  | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | | Перечислены основы для формирования компетенции ПК-1:  - Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов, ограничений УК-2  - Способен использовать современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности ОПК-2 | |
|  | Средства и технологии оценки | | * итоговая аттестация | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Наименование компетенции | | УК1 | |
| 2. | Указание типа компетенции | Универсальная |  | |
| 3 | Определение, содержание, основные сущностные характеристики компетенции | | Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач. Компетенция предполагает способность человека ставить себе образовательные цели под возникающие жизненные задачи, подбирать способы решения и средства развития (в том числе с использованием цифровых средств) других необходимых компетенций. | |
| 4 | Дескриптор знаний, умений и навыков по уровням | | Уровни сформированности компетенции обучающегося | Индикаторы |
|  | У выпускника должны быть сформированы следующие компетенции на начальном уровне:  Знать: основы поиска, критического анализа и синтеза информации,  системного подхода для решения поставленных задач  Уметь: применять основы поиска, критического анализа и синтеза  информации, системного подхода для решения поставленных задач  Владеть: навыками применения основ поиска, критического анализа  и синтеза информации, системного подхода для решения поставленных задач | | Начальный | Знает принципы работы с сетью.  Уметь писать программы на языке JAVA  Уметь управлять большими объемами данных и быстро получать нужное с помощью запросов;  Уметь тестировать код;  Знать методы отладки и тестирования;  Уметь находить и исправлять даже мельчайшие ошибки в своём или чужом коде.  Уметь работать с сетью |
|  | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | | Перечислены основы для формирования компетенции УК-1:  Для формирования данной компетенции необходимо обладать следующими компетенциями на начальном уровне:  - УК-2,  На базовом уровне:  ОПК-2, ОПК-8, ОПК-9 | |
|  | Средства и технологии оценки | | * итоговая аттестация | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Наименование компетенции | | УК2 | |
| 2. | Указание типа компетенции | Универсальная |  | |
| 3 | Определение, содержание, основные сущностные характеристики компетенции | | Способность определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений. | |
| 4 | Дескриптор знаний, умений и навыков по уровням | | Уровни сформированности компетенции обучающегося | Индикаторы |
|  | У выпускника должны быть сформированы следующие компетенции на начальном уровне:  Знать: основы определения круга задач в рамках поставленной цели  и выбора оптимальных способов их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений.  Уметь: применять основы определения круга задач в рамках постав-  ленной цели и выбора оптимальных способов их решения, исходя из  действующих правовых норм, имеющихся ресурсов и ограничений.  Владеть: навыками применения основ определения круга задач в рамках поставленной цели и выбора оптимальных способов их решений, исходя из действующих правовых норм, имеющихся ресурсов и  ограничении | | Начальный | Умеет анализировать альтернативные варианты решений для достижения намеченных результатов; разрабатывать план, определять целевые этапы и основные направления работ.  Знает, как работать с распределенной системой контроля версий, и сможете вносить или откатывать изменения в проекте одновременно с другими разработчиками;  Умеет определять круг задач и последовательность их исполнения в рамках написания программного кода под конкретную задачу.  Может выбрать оптимальное решение и сценарий написания кода из предложенных вариантов. |
|  | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | | Перечислены основы для формирования компетенции УК-2:  Для формирования данной компетенции необходимо обладать следующими компетенциями на начальном уровне:  - УК-1,  На базовом уровне:  ОПК-2, ОПК-8, ОПК-9 | |
|  | Средства и технологии оценки | | * итоговая аттестация | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Наименование компетенции | | ОПК2 | |
| 2. | Указание типа компетенции | Общепрофессиональная |  | |
| 3 | Определение, содержание, основные сущностные характеристики компетенции | | Способность использовать современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности | |
| 4 | Дескриптор знаний, умений и навыков по уровням | | Уровни сформированности компетенции обучающегося | Индикаторы |
|  | У выпускника должны быть сформированы следующие компетенции на базовом уровне:  Знать: основы применения  математического аппарата, методов анализа и моделирования, теоретического и экспериментального исследования при решении профессиональных задач.  Уметь: применять соответствующий физико-математический аппарат, методы анализа и моделирования, теоретического и экспериментального исследования при решении профессиональных задач  Владеть:  Навыками математического аппарата, методов анализа и моделирования, теоретического и экспериментального исследования при решении профессиональных задач | | Базовый | Знает принципы понятие цикл (while, for in), одномерный и двумерный массив, сортировка, стек, очередь, список;  Знает основные концепции программирования сложных алгоритмов;  Знает методы работы со строками; типы сложных сортировок. |
|  | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | | Перечислены основы для формирования компетенции ОПК-2:  Для формирования данной компетенции необходимо обладать следующими компетенциями на начальном уровне:  - УК-1, УК-2  На базовом уровне:  ОПК-8, ОПК-9 | |
|  | Средства и технологии оценки | | * итоговая аттестация | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Наименование компетенции | | ОПК8 | |
| 2. | Указание типа компетенции | Общепрофессиональная |  | |
| 3 | Определение, содержание, основные сущностные характеристики компетенции | | Способен разрабатывать алгоритмы и программы, пригодные для практического применения | |
| 4 | Дескриптор знаний, умений и навыков по уровням | | Уровни сформированности компетенции обучающегося | Индикаторы |
|  | У выпускника должны быть сформированы следующие компетенции на базовом уровне:  Знать: основные языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий.  Уметь: применять языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.  Владеть: навыками программирования, отладки и тестирования прототипов программно-технических комплексов | | Базовый | Знает принципы построения алгоритмов с применением языка JAVA;  Умеет решать сложные алгоритмические задачи с использованием одномерных и двумерных массивов;  Умеет использовать принципы сортировок на языке программирования Java;  Может тестировать свой и чужой код, находить и исправлять даже мельчайшие ошибки. |
|  | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | | Перечислены основы для формирования компетенции ОПК-8:  Для формирования данной компетенции необходимо обладать следующими компетенциями на начальном уровне:  - УК-1, УК-2  На базовом уровне:  ОПК-2, ОПК-9 | |
|  | Средства и технологии оценки | | * итоговая аттестация | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Наименование компетенции | | ОПК9 | |
| 2. | Указание типа компетенции | Общепрофессиональная |  | |
| 3 | Определение, содержание, основные сущностные характеристики компетенции | | Способен осваивать методики использования программных средств для решения практических задач. | |
| 4 | Дескриптор знаний, умений и навыков по уровням | | Уровни сформированности компетенции обучающегося | Индикаторы |
|  | У выпускника должны быть сформированы следующие компетенции на базовом уровне:  Знать:  классификацию программных средств и возможности их применения для решения практических задач.  Уметь:  находить и анализировать техническую документацию по использованию программного средства, выбирать и использовать необходимые функции программных средств для решения конкретной задачи:  Владеть:  способами описания методики использования программного средства для решения конкретной задачи в виде технической документации. | | Базовый | Знает основные фрейм-ворки для языка программирования JAVA.  Умеет формально выполнять алгоритмы, описанные с использованием простых и сложны сортировок, одномерных и двумерных массивов, циклов;  Умеет использовать готовые прикладные компьютерные программы и сервисы;  Может разработать техническую документацию по разработке программного кода и его тестирования. |
|  | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | | Перечислены основы для формирования компетенции ОПК-9:  Для формирования данной компетенции необходимо обладать следующими компетенциями на начальном уровне:  - УК-1, УК-2  На базовом уровне:  ОПК-2, ОПК-9 | |
|  | Средства и технологии оценки | | * итоговая аттестация | |

1. **Рекомендаций к программе от работодателей:**

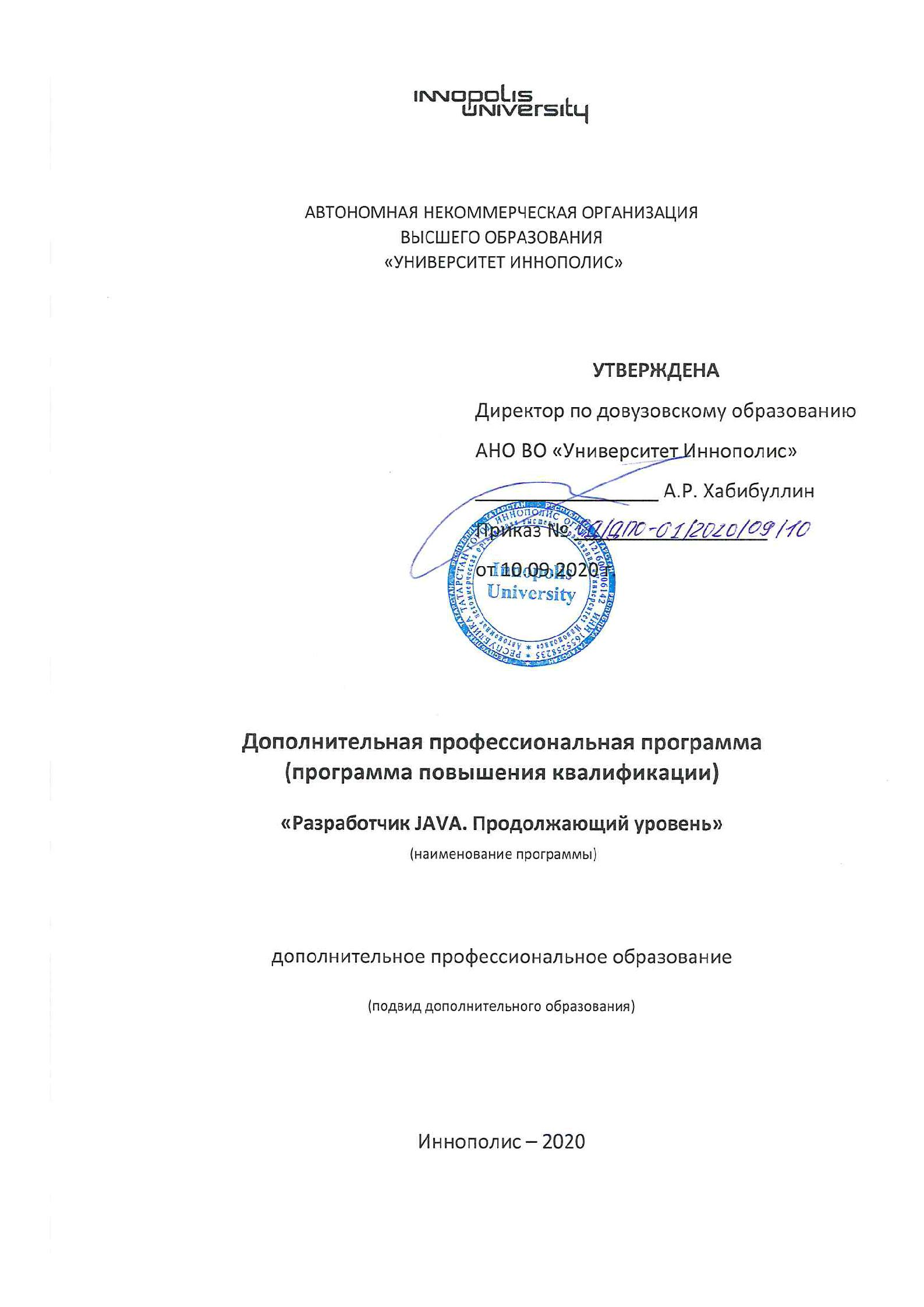
В приложении к паспорту программы

1. **Возможные сценарии профессиональной траектории граждан по итогам освоения образовательной программы «Разработчик JAVA. Продолжающий уровень»**

|  |  |
| --- | --- |
| **Цели получения персонального цифрового сертификата** | |
| **текущий статус** | **цель** |
| **Трудоустройство** | |
| Стоящий на учете в Центре занятости  Безработный | Трудоустройство по направлению разработчик программного обеспечения на языке JAVA.  Освоение программы подразумевает несколько вариантов трудоустройства:  Разработчик программного обеспечения на языке JAVA. Уровень Junior и Middle Developer.  В классификаторе ИТ-компаний по типу продукта возможны следующие сферы:  - Мобильная разработка  - Разработка игр  - «Коробочные решения»  Тестировщик программного обеспечения.  Знания и навыки, полученные на курсе, позволяют понимать структуру кода и архитектуру решаемой задачи. Знания языка Java на уровне middle позволяет существенно ускорять процесс постановки задач разработчикам и минимизировать недопонимание между тестировщиком и разработчиком. Тестировщик данного уровня в основном с автотестами, разработанными на основе знаний языка Java. Знание языка программирования Java дает возможность перехода с тестирования на разработку уровня Junior.  WEB разработчик.  Знания языка программирования JAVA на уровне middle позволяет web-разработчику совершенствовать свои навыки и компетенции в Java Script, что дает возможность решать более сложные задачи при разработке web-приложений и web-страниц, а также появляется возможность построения более сложных архитектур, что упрощает и совершенствует процесс web-разработки различного уровня.  Мобильная разработка  Знание языка на уровне middle позволяет систематизировать знания и самостоятельно решать задачи при разработке приложений. Так же дает возможность вести группу Junior по проекту, помогая при решении простых задач. Так же разработчик со знаниями языка на данном уровне знает весь стек технологий, применяемый для решений той или иной задачи в работе над проектом.  Самозанятый (фриланс) – удаленная разработка программного обеспечения (разработка мобильных приложений).  Курс, освоенный на уровне Middle позволяет самостоятельно заниматься разработкой небольших проектов, включая в работу junior разработчиков. Позволяет создавать более эффективный и простой код, избегая простых ошибок/багов. |
| безработный по состоянию здоровья |
| **Развитие компетенций в текущей сфере занятости** | |
| Работающий по найму в организации, на предприятии | Сохранение текущего рабочего места: развитие дополнительных профессиональных компетенций по разработке на языке JAVA позволит работающим по найму сотрудникам актуализировать знания по разработке, освоить новый стек технологий и соответствовать требованиям цифровой экономики к кадрам. |
| Работающий по найму в организации, на предприятии | Прохождение обучения способствует развитию следующих профессиональных качеств:  Приобретение навыков работы с кодом.  Возможность самостоятельно писать простые программы, необходимые для текущей деятельности. Освоение методов и средств проектирования программных интерфейсов.  Работа с базами данных. Знание методов проектирования и обработки баз данных.  Разработчикам, программирующим на других языках программирования, освоение языка JAVA позволит разрабатывать кросс-платформенные приложения, позволит освоить новые технологии и фреймворки. Знание дополнительно еще одного или нескольких языков программирования на уровне middle позволяет заниматься грамотными оптимизацией, рефакторингом и обзором кода. Знание языка Java на уровне middle позволяет эффективно оценивать затраты на ту или иную задачу и дает возможность проверки решения одной задачи на разных языках программирования, выбрав оптимальный вариант. |
| Работающий по найму в организации, на предприятии | Повышение заработной платы: развитие компетенций по новому стеку технологий, повышение универсальных, общепрофессиональных и профессиональных компетенций обеспечивает сотруднику возможность оптимизировать собственные рабочие процессы, а также влиять на оптимизацию процессов на предприятии. Что может значительно экономить затраты и увеличивать прибыль от продаж. Исходя из этого, специалист, прошедший обучение «дорожает» на рынке труда и может рассчитывать на увеличение заработной платы на текущем предприятии. |
| Работающий по найму в организации, на предприятии | Смена работы без изменения сферы профессиональной деятельности: переход в более крупную компанию или компанию, расположенную в другом регионе или внутри компании с повышением заработной платы. |
| временно отсутствующий на рабочем месте (декрет, отпуск по уходу за ребенком и др.) | Прохождение обучения по программе способствует сохранению и развитию квалификации. Обеспечивает возможность соответствовать должностным требованиям в условиях быстро меняющейся цифровой среды. |
| **Переход в новую сферу занятости** | |
| освоение новой сферы занятости | Прохождение обучения по программе способствует расширению кругозора. Самозанятым гражданам освоение нового стека технологий даст возможность выполнять заказы по разработке мобильных приложений и web элементов для сторонних заказчиков. |
| освоение смежных профессиональных областей |  |

**7.Приложенные Скан-копии**



**I. Общие положения**

1. Дополнительная профессиональная программа «Разработчик JAVA. Продолжающий уровень» разработана в соответствии с нормами Федерального закона от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации», с учетом требований приказа Минобрнауки России от 01 июля 2013 г. № 499 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам», с изменением внесенным приказом Минобрнауки России от 15 ноября 2013 г. № 1244 «О внесении изменений в Порядок организации и осуществления образовательной деятельности по дополнительным профессиональным программам, утвержденный приказом Министерства образования и науки Российской Федерации от 01 июля 2013 г. № 499», приказа Минобрнауки России от 23.08.2017 № 816 «Об утверждении Порядка применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ», федерального государственного образовательного стандарта высшего образования по направлению подготовки 09.03.01. Информатика и вычислительная техника (уровень бакалавриат), утвержденного приказом Минобрнауки России от 19 сентября 2017 г. № 929; (далее вместе – ФГОС ВО), а также профессионального стандарта «Программист», утвержденного приказом Министерства труда и социальной защиты РФ от 18 ноября 2013 г. № 679н.

2. Повышение квалификации заинтересованных лиц (далее – слушатели), осуществляемое в соответствии с Программой (далее – подготовка), проводится в АНО ВО «Университет Иннополис» (далее – Университет) в соответствии с учебным планом в очной форме обучения с применением электронного обучения и дистанционных образовательных технологий. Уровень подготовки слушателей к освоению программы определяется входным тестированием (задания входного тестирования приведены в Приложении к настоящей программе).

3. Разделы, включенные в учебный план Программы, используются для последующей разработки календарного учебного графика, учебно-тематического плана, рабочей программы, оценочных и методических материалов. Перечисленные документы разрабатываются Университетом самостоятельно, с учетом актуальных положений законодательства об образовании, законодательства в области информационных технологий и смежных областей знаний, ФГОС ВО, профессионального стандарта «Программист».

4. Программа регламентирует требования к повышению квалификации обучающихся за счет приобщения их к новым технологиям, способным помочь им в реализации собственного потенциала, принципам программирования и применения их при решении алгоритмических задач и создании графических проектов на языке программирования Java с использованием вспомогательных библиотек.

Срок освоения Программы составляет **72 академических часа**.

К освоению Программы допускаются физические лица, имеющие или получающие высшее образование или среднее профессиональное образование.

5. Область профессиональной деятельности:

Разработка программного обеспечения

**II. Цель подготовки**

6. Целью подготовки слушателей по Программе является повышение квалификации обучающихся за счет создания условий для их саморазвития через решение алгоритмических задач с использованием одномерных и двумерных массивов, строк, сортировок.

**III. Характеристика новой и/или повышаемой квалификации и связанных с ней видов профессиональной деятельности, трудовых функций и (или) уровней квалификации**

7. Виды профессиональной деятельности, трудовая функция**,** требуемые знания и умения, указанные в профессиональном стандарте «Программист», утвержденном приказом Министерства труда и социальной защиты РФ от 18 ноября 2013 г. № 679н представлены в таблице ниже:

**Характеристика новой и/или повышаемой квалификации, связанной с видом профессиональной деятельности и трудовыми функциями в соответствии с профессиональным стандартом «Программист»**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Область профессиональной деятельности** | **Тип задач профессиональной деятельности** | **Код и наименование профессиональной компетенции** | **Код и наименование индикатора достижения профессиональной компетенции выпускника** | **Трудовые действия** | **Трудовая функция** | **Обобщенная трудовая функция** | **Вид профессиональной деятельности** |
| Разработка программного обеспечения | Проектный | **ПК-1:** Способность применять методы и средства проектирования при разработке программных интерфейсов, с учетом методов, средств и структур данных | **Знать:**  Методы и средства проектирования программного обеспечения;  - Методы и средства проектирования баз данных;  - Методы и средства проектирования программных интерфейсов.  **Уметь:**  - Использовать существующие типовые решения и шаблоны проектирования программного обеспечения;  - Применять методы и средства проектирования программного обеспечения, структур данных, баз данных, программных интерфейсов | Проектирование структур данных;  - Проектирование баз данных;  - Проектирование программных интерфейсов | Проектирование программного обеспечения | Разработка требований и проектирование программного обеспечения | Разработка программного обеспечения |

**IV. Характеристика компетенций, подлежащих совершенствованию, и (или) перечень новых компетенций, формирующихся в результате освоения программы**

8. В ходе освоения Программы слушателем совершенствуются следующие **универсальные компетенции**:

УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач;

УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений.

В ходе освоения Программы слушателем совершенствуются следующие **общепрофессиональные компетенции**:

ОПК-8. Способен разрабатывать алгоритмы и программы, пригодные для практического применения;

ОПК-9. Способен осваивать методики использования программных средств для решения практических задач.

В ходе освоения Программы слушателем совершенствуются следующие **профессиональные компетенции**:

ПК-1. Способность применять методы и средства проектирования при разработке программных интерфейсов, с учетом методов, средств и структур данных

9. Карта компетенции раскрывает компонентный состав компетенции, технологии ее формирования и оценки:

1) Дисциплинарная карта компетенции УК-1.

|  |  |
| --- | --- |
| УК-1: способность осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач | |
| Технологии формирования: | Средства и технологии оценки: |
| Лекция, практические занятия, самостоятельная работа | Текущий контроль  Итоговая аттестация |

2) Дисциплинарная карта компетенции УК-2.

|  |  |
| --- | --- |
| УК-2: способность определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений | |
| Технологии формирования: | Средства и технологии оценки: |
| Лекция, практические занятия, самостоятельная работа | Текущий контроль  Итоговая аттестация |

3) Дисциплинарная карта компетенции ОПК-8.

|  |  |
| --- | --- |
| ОПК-8: способность Способен разрабатывать алгоритмы и программы, пригодные для практического применения | |
| Технологии формирования: | Средства и технологии оценки: |
| Лекция, практические занятия, самостоятельная работа | Текущий контроль  Итоговая аттестация |

4) Дисциплинарная карта компетенции ОПК-9.

|  |  |
| --- | --- |
| ОПК-9: способность осваивать методики использования программных средств для решения практических задач | |
| Технологии формирования: | Средства и технологии оценки: |
| Лекция, практические занятия, самостоятельная работа | Текущий контроль  Итоговая аттестация |

5) Дисциплинарная карта компетенции ПК-1

|  |  |
| --- | --- |
| ПК-1: способность применять методы и средства проектирования при разработке программных интерфейсов, с учетом методов, средств и структур данных | |
| Технологии формирования: | Средства и технологии оценки: |
| Лекция, практические занятия, самостоятельная работа | Текущий контроль  Итоговая аттестация |

**V. Планируемые результаты обучения по ДПП**

10. Результатами подготовки по Программе является повышение уровня профессиональной квалификации слушателей за счет изучения программирования с использованием одномерных и двумерных массивов, строк, сортировок

11. В результате освоения Программы продолжающего уровня слушатель должен:

**знать:**

* понятие цикл (while, for in), одномерный и двумерный массив, сортировка, стек, очередь, список;
* основные концепции программирования сложных алгоритмов; методы работы со строками;
* типы сложных сортировок.

**уметь:**

* решать сложные алгоритмические задачи с использованием одномерных и двумерных массивов, сортировок на языке программирования Java;
* использовать списки;
* формально выполнять алгоритмы, описанные с использованием простых и сложны сортировок, одномерных и двумерных массивов, циклов; использовать готовые прикладные компьютерные программы и сервисы;
* создавать и выполнять программы для решения сложных алгоритмических задач в программе Java;

**иметь навыки:**

* работы с персональным компьютером; самостоятельного построения компьютерной модели;
* беглого компьютерного ввода на русском и английском языках;
* свободного владения компьютерным графическим интерфейсом;
* применения сложных алгоритмов при решении задач на языке Java;
* использования теории алгоритмов при решении олимпиадных задач;
* выбора способа представления данных в зависимости от постановленной задачи.

**VI. Организационно-педагогические условия реализации Программы**

12. Реализация Программы должна обеспечить совершенствование компетенций обучающихся в условиях инновационного процесса цифровизации общества за счет освоения ими методов, средств и структур данных проектирования программных интерфейсов при решении алгоритмических задач с использованием сложных структур данных; изучения программирования посредством одномерных и двумерных массивов, строк, сортировок.

13. Учебный процесс организуется с применением инновационных технологий и методик обучения с использованием электронного обучения и дистанционных образовательных технологий, способных обеспечить получение слушателями знаний, умений и навыков в области дополнительного образования.

14. Реализация Программы обеспечивается научно-педагогическими кадрами Университета, допустимо привлечение к образовательному процессу высококвалифицированных специалистов из числа руководителей и ведущих специалистов государственных учреждений, а также преподавателей ведущих российских и иностранных образовательных организаций.

**VII. Учебный план ДПП**

15. Объем Программы составляет **72 академических часа**.

16. Учебный план Программы определяет перечень, последовательность, общую трудоемкость разделов и формы контроля знаний.

Образовательная деятельность слушателей предусматривает следующие виды учебных занятий и учебных работ:

- лекции;

- практические занятия;

- самостоятельная работа;

- итоговую аттестацию (в форме зачета).

**Учебный план программы повышения квалификации  
«Разработчик JAVA. Продолжающий уровень»**

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование раздела (модуля)** | **Общая**  **трудоемкость** | **Форма контроля** |
| 1. | Условные операторы | 4 | текущий контроль |
| 2. | Циклы | 4 | текущий контроль |
| 3. | Одномерные массивы | 8 | текущий контроль |
| 4. | Простые сортировки | 6 | текущий контроль |
| 5. | Сортировка подсчетом | 8 | текущий контроль |
| 6. | Двумерные массивы | 8 | текущий контроль |
| 7. | Простые структуры данных. Списки, стеки, очереди | 8 | текущий контроль |
| 8. | Работа со строками | 8 | текущий контроль |
| 9. | Сложные сортировки. | 8 | текущий контроль |
| 10. | Индивидуальные проекты | 8 | текущий контроль |
| 11. | Итоговая аттестация | 2 | |
| **Итого:** | | **72** | |

17. Матрица соотнесения разделов учебного плана Программы и формируемых в них универсальных, общепрофессиональных и профессиональных компетенций:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Наименование раздела** | **Всего часов** | **Компетенции** | | | | |
| **универсальные** | | **общепрофессиональные** | | **профессиональные** |
| **УК-1** | **УК-2** | **ОПК-8** | **ОПК-9** | **ПК-1** |
| 1. | Условные операторы | 4 | + | + | + | + | + |
| 2. | Циклы | 4 |  |  | + | + | + |
| 3. | Одномерные массивы | 8 |  | + | + | + | + |
| 4. | Простые сортировки | 6 | + | + | + | + | + |
| 5. | Сортировка подсчетом | 8 | + | + | + | + | + |
| 6. | Двумерные массивы | 8 | + | + | + | + | + |
| 7. | Простые структуры данных. Списки, стеки, очереди | 8 | + | + | + | + | + |
| 8. | Работа со строками | 8 | + | + | + | + | + |
| 9. | Сложные сортировки. | 8 | + | + | + | + | + |
| 10. | Индивидуальные проекты | 8 | + | + | + | + | + |
| 11. | Итоговая аттестация | 2 | + | + | + | + | + |

**VIII. Календарный учебный график**

18. Календарный учебный график представляет собой график учебного процесса, устанавливающий последовательность и продолжительность обучения и итоговой аттестации по учебным дням.

**Календарный учебный график программы повышения квалификации  
«Разработчик JAVA. Продолжающий уровень»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№**  **п/п** | **Наименование раздела** | **Учебные дни** | | | | |
| 1-7 | 8-14 | 15-21 | 22-28 | 29-30 |
| 1. | Условные операторы | + |  |  |  |  |
| 2. | Циклы | + |  |  |  |  |
| 3. | Одномерные массивы | + | + |  |  |  |
| 4. | Простые сортировки |  | + |  |  |  |
| 5. | Сортировка подсчетом |  | + | + |  |  |
| 6. | Двумерные массивы |  |  | + |  |  |
| 7. | Простые структуры данных. Списки, стеки, очереди |  |  | + | + |  |
| 8. | Работа со строками |  |  |  | + |  |
| 9. | Сложные сортировки. |  |  |  | + | + |
| 10. | Индивидуальные проекты |  |  |  |  | + |
| 11. | Итоговая аттестация |  |  |  |  | + |

**I**X**. Рабочая программа учебных предметов, курсов, дисциплин**

**(модулей)**

19. Рабочая программа содержит перечень разделов и тем, а также рассматриваемых в них вопросов с учетом их трудоемкости.

Рабочая программа разрабатывается Университетом с учетом актуальных положений законодательства об образовании и смежных областей знаний; ФГОС ВО, профессионального стандарта «Программист». Программа реализуется с применением электронного обучения и дистанционных образовательных технологий. Лекционные и практические занятия проводятся в форме вебинаров. Задания и материалы программы размещаются в системе LMS Moodle, доступ к которой получают все обучающиеся.

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Наименование и краткое содержание раздела** | **Объем, часов** |
| 1. | **Условные операторы.**  *Краткое содержание:* Простые, операторы сравнения; условные операторы if, if/else; решение задач с использованием условных операторов и операторов сравнения  *Лекции****:*** Простые, операторы сравнения; условные операторы if, if/else  *Практические занятия:* Простые, операторы сравнения; условные операторы if, if/else;решение задач с использованием условных операторов и операторов сравнения | 4 |
| 2. | **Циклы.**  *Краткое содержание:* Циклы while, for in; бесконечные циклы; вложенные циклы; решение задач на составление простых алгоритмов.  *Лекции:* Циклы while, for in;  *Практические занятия*: Циклы while, for in; бесконечные циклы; вложенные циклы; решение задач на составление простых алгоритмов. | 4 |
| 3. | **Одномерные массивы.**  *Краткое содержание:* Основные операторы программирования; Понятие структуры данных – массив. Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике*.*  *Лекции:* Основные операторы программирования.  *Практические занятия*: Основные операторы программирования; Понятие структуры данных – массив. Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике. | 8 |
| 4. | **Простые сортировки.**  *Краткое содержание: С*ортировка пузырьком; сортировка вставками; способы реализации сортировок; время сортировок; функция сортировки пузырьком; решение задач с использованием сортировок пузырьком и вставками.  *Лекции:* сортировка пузырьком; сортировка вставками; способы реализации сортировок; время сортировок; функция сортировки пузырьком  *Практические занятия:* сортировка пузырьком; сортировка вставками; способы реализации сортировок; время сортировок; функция сортировки пузырьком; решение задач с использованием сортировок пузырьком и вставками. | 6 |
| 5. | **Сортировка подсчетом.**  *Краткое содержание:* Сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки; решение задач с использованием сортировки подсчетом.  *Лекции:* сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки.  *Практические занятия:* сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки; решение задач с использованием сортировки подсчетом. | 8 |
| 6. | **Двумерные массивы.**  *Краткое содержание: П*онятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы; решение задач на двумерные массивы и матрицы.  *Лекции****:*** понятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы.  *Практические занятия:* понятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы; решение задач на двумерные массивы и матрицы. | 8 |
| 7. | **Простые структуры данных. Списки, стеки, очереди.**  *Краткое содержание: Ф*ормирование строк; строковые функции и методы, операции над строками; обработка строк; использование строк при решении задач и создании проектов  *Лекции:* формирование строк; строковые функции и методы, операции над строками; обработка строк  *Практические занятия:* формирование строк; строковые функции и методы, операции над строками; обработка строк; использование строк при решении задач и создании проектов | 8 |
| 8. | **Работа со строками.**  *Краткое содержание:* формирование строк; строковые функции и методы, операции над строками; обработка строк; использование строк при решении задач и создании проектов  *Лекции:* формирование строк; строковые функции и методы, операции над строками; обработка строк  *Практические занятия:* формирование строк; строковые функции и методы, операции над строками; обработка строк; использование строк при решении задач и создании проектов | 8 |
| 9. | **Сложные сортировки.**  *Краткое содержание:* Быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок. рассматривается использование сортировок при решении задач и создании проектов  *Лекции:* быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок.  *Практические* занятия: быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок. рассматривается использование сортировок при решении задач и создании проектов | 8 |
| 10. | **Индивидуальные проекты.**  *Краткое содержание:*Закрепление пройденного материала на примере создания собственного проекта. Обучение навыкам создания проектов и работа с большим участком кода.  *Практические занятия*: Закрепление пройденного материала на примере создания собственного проекта. Обучение навыкам создания проектов и работа с большим участком кода. | 8 |
| 11. | **Итоговая аттестация** | 2 |

20. Учебно-тематический план Программы определяет тематическое содержание, последовательность разделов и (или) тем и их трудоемкость.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Наименование раздела** | **Количество часов** | | | |
| **Общая трудоемкость** | **в том числе** | | |
| **Лекции** | **Практические занятия** | **Самостоятельная работа** |
| 1. **Условные операторы** | | **4** | **1** | **2** | **1** |
| 1.1. | Простые, операторы сравнения; условные операторы if, if/else  Решение задач с использованием условных операторов и операторов сравнения | 4 | 1 | 2 | 1 |
| 1. **Циклы** | | **4** | **1** | **2** | **1** |
| 2.1. | Циклы while, for in;  Бесконечные циклы; вложенные циклы | 2 | 1 | 1 |  |
| 2.2. | Решение задач на составление простых алгоритмов | 2 |  | 1 | 1 |
| 1. **Одномерные массивы** | | **8** | **1** | **5** | **2** |
| 3.1. | *Основные операторы программирования.*  *Понятие структуры данных – массив.* | 4 | **1** | **2** | **1** |
| 3.2. | *Объявление, заполнение и действия над массивами. Использование массивов в компьютерной графике.* | 4 |  | **3** | **1** |
| 1. **Простые сортировки.** | | **6** | **1** | **4** | **1** |
| 4.1. | Сортировка пузырьком; сортировка вставками | 2 | 1 | 1 |  |
| 4.2. | Способы реализации сортировок; время сортировок; функция сортировки пузырьком | 1 |  | 1 |  |
| 4.3. | Решение задач с использованием сортировок пузырьком и вставками. | 3 |  | 2 | 1 |
| 1. **Сортировка подсчетом** | | **8** | **1** | **4** | **3** |
| 5.1. | *Сортировка подсчётом (Counting Sort), способы реализации сортировки, время сортировки.* | 4 | 1 | 2 | 1 |
| 5.2. | Решение задач с использованием сортировки подсчетом. | 4 |  | 2 | 2 |
| 1. **Двумерные массивы.** | | **8** | **1** | **6** | **1** |
| 6.1. | Понятие двумерного массива; заполнение двумерного массива; вывод двумерного массива на экран; считывание двумерного массива с клавиатуры; обработка элементов двумерного массива; понятие матрицы; способы вывода матрицы на экран; инициализация элементов матрицы. | 4 | 1 | 3 |  |
| 6.2. | Решение задач на двумерные массивы и матрицы. | 4 |  | 3 | 1 |
| 1. **Простые структуры данных. Списки, стеки, очереди.** | | **8** |  | **6** | **2** |
| 7.1. | Формирование строк; строковые функции и методы, операции над строками; обработка строк  Использование строк при решении задач и создании проектов | 8 |  | 6 | 2 |
| 1. **Работа со строками.** | | **8** |  | **6** | **2** |
| 8.1. | Формирование строк; строковые функции и методы, операции над строками; обработка строк | 4 |  | 3 | 1 |
| 8.2. | Использование строк при решении задач и создании проектов | 4 |  | 3 | 1 |
| 1. **Сложные сортировки.** | | **8** |  | **6** | **2** |
| 9.1. | Быстрая сортировка; сортировка слиянием; принципы работы сортировок; реализация сортировок; оптимизация сортировок. | 4 |  | 3 | 1 |
| 9.2. | Использование сортировок при решении задач и создании проектов | 4 |  | 3 | 1 |
| 1. **Индивидуальные проекты.** | | **8** |  | **6** | **2** |
| 10.1. | *Закрепление пройденного материала на примере создания собственного проекта.* | 2 |  | **2** |  |
| 10.2. | *Обучение навыкам создания проектов и работа с большим участком кода.* | 6 |  | **4** | **2** |
| 11. | **Итоговая аттестация** | **2** | | | |
|  | **Итого:** | **72** | | | |

**X. Формы аттестации**

21. Слушатели, успешно выполнившие все элементы учебного плана, допускаются к итоговой аттестации. Итоговая аттестация по Программе проводится в форме контрольного тестирования, состоящего из задач, выбранных случайным образом из общего объема изученного материала. Контрольный тест должен быть выполнен в объеме более 60 % от общего объема задач. Задания для проведения промежуточной аттестации формируются в соответствии с уровнем знаний обучающихся.

22. Лицам, успешно освоившим Программу и прошедшим итоговую аттестацию, выдается удостоверение о повышении квалификации установленного образца.

При освоении Программы параллельно с получением высшего образования удостоверение о повышении квалификации выдается одновременно с получением соответствующего документа об образовании и квалификации.

23. Лицам, не прошедшим итоговую аттестацию или получившим на итоговой аттестации неудовлетворительные результаты, а также лицам, освоившим часть Программы и (или) отчисленным из Университета, выдается справка об обучении или о периоде обучения по образцу, самостоятельно устанавливаемому Университетом.

**XI. Оценочные материалы**

24. Контроль знаний, полученных слушателями при освоении разделов (модулей) Программы, осуществляется в следующих формах:

* контроль знаний (входное тестирование) – оценивает уровня знаний слушателя, необходимых для освоения разделов Программы;
* текущий контроль – самостоятельная оценка изученного материала, необходимого для перехода к следующим разделам Программы;
* итоговая аттестация (контрольное тестирование) – оценивает завершает изучение всей Программы.

Итоговая аттестация проводится в форме выполнения заданий контрольного теста, демонстрирующего освоение слушателем всех изученных разделов Программы.

25. В ходе освоения Программы каждый слушатель выполняет задания, разработанные преподавателем. Примеры заданий представлены в следующей таблице:

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование раздела (модуля)** | **Задание** | **Критерии оценки** |
| 1. | Условные операторы. | Решить задачи. Предложенные преподавателем, с помощью условных операторов (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме из предложенных 5, с использованием условных операторов. |
| 2. | Циклы. | Решить задачи. Предложенные преподавателем, с помощью циклических операторов (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме из предложенных 5, с использованием циклов. |
| 3. | Одномерные массивы. | Решить задачи. Предложенные преподавателем, используя одномерные массивы (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме с использованием одномерных массивов. |
| 4. | Простые сортировки. | Решить задачи, предложенные преподавателем, используя простые сортировки (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме из предложенных 5, с использованием различных сортировок. |
| 5. | Сортировка подсчетом. | Решить задачи, предложенные преподавателем, используя сортировку подсчетом (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме из предложенных 5, с сортировки подсчетом. |
| 6. | Двумерные массивы. | Решить, предложенные преподавателем задачи на двумерные массивы, матрицы. (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме из предложенных 5, используя двумерные массивы. |
| 7. | Простые структуры данных. Списки, стеки, очереди. | Решить задачи. Предложенные преподавателем, применив стеки и очереди (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме с использованием простых структур данных. |
| 8. | Работа со строками. | Решить задачи. Предложенные преподавателем, используя различные варианты строк (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме с использованием строк. |
| 9. | Сложные сортировки. | Решить задачи. Предложенные преподавателем, используя сложные сортировки (Приложение 2). | Слушатель должен решить не менее 3 задач по данной теме с использованием сложных сортировок. |
| 10. | Индивидуальные проекты. | Разработать индивидуальный проект в соответствии с выбранной темой. | Слушатель должен разработать собственный проект |
| 11. | Итоговая аттестация  Контрольный тест | Написать проект, который представляет собой программу, позволяющую стабильно работать в сети Интернет. Примерная тематика:  1. Развлекательная программа (игра, анимация, клип и т. п.)  2. Мобильное приложение | Защита итоговой проектной работы, написанной на языке Java, стабильно работающей в сети Интернет.  Успешное решение более 60% контрольного теста |

**26. Перечень примерных практических заданий:**

Раздел 1. Условные операторы

**Задача 1. Координатные четверти.**

Даны координаты двух точек на плоскости, требуется определить, лежат ли они в одной координатной четверти или нет (все координаты отличны от нуля).

**Входные данные**

Вводятся 4 числа: координаты первой точки (*x1*, *y1*) и координаты второй точки (*x2*, *y2*).

**Выходные данные**

Программа должна вывести слово YES, если точки находятся в одной координатной четверти, в противном случае вывести слово NO.

**Примеры**

**входные данные**

3

3

5

1

**выходные данные**

YES

**Задача 2. Максимум из 3-х.**

**Входные данные**

Даны три целых числа, каждое записано в отдельной строке.

**Выходные данные**

Выведите наибольшее из данных чисел (программа должна вывести ровно одно целое число).

**Примеры**

**входные данные**

1

2

3

**выходные данные**

3

**Задача 3. Количество равных из 3-х.**

**Входные данные**

Даны три целых числа, записанных в отдельных строках. Определите, сколько среди них совпадающих.

**Выходные данные**

Программа должна вывести одно из чисел: 3 (если все совпадают), 2 (если два совпадают) или 0 (если все числа различны).

**Примеры**

**входные данные**

1

2

2

**выходные данные**

2

**Задача 4. Четные и нечетные числа.**

Даны три целых числа *A*, *B*, *C*. Определить, есть ли среди них хотя бы одно четное и хотя бы одно нечетное.

**Входные данные**

Числа *A*, *B*, *C*, не превышающие по модулю 10000.

**Выходные данные**

Одна строка – "YES" или "NO".

**Примеры**

**входные данные**

3

4

5

**выходные данные**

YES

**входные данные**

7

7

7

**выходные данные**

NO

**Задача 5. Упорядочить 3 числа.**

Дано три числа, записанный в отдельных строках. Упорядочите их в порядке неубывания. Программа должна считывать три числа a, b, c, затем программа должна менять их значения так, чтобы стали выполнены условия a <= b <= c, затем программа выводит тройку a, b, c.

**Входные данные**

Вводятся три числа, каждое записано в отдельной строке.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

1

2

1

**выходные данные**

1 1 2

Раздел 2. Циклы.

**Задача 6. Четные числа.**

**Входные данные**

Вводятся целые числа *a* и *b*. Гарантируется, что *a* не превосходит *b*

**Выходные данные**

Выведите (через пробел) все четные числа от *a* до *b* (включительно).

**Примеры**

**входные данные**

2

5

**выходные данные**

2 4

**Задача 7. Остаток.**

**Входные данные**

Вводятся 4 числа: *a*, *b*, *c* и *d*.

**Выходные данные**

Выведите все числа на отрезке от *a* до *b*, дающие остаток *c* при делении на *d*. Если таких чисел не существует, то ничего выводить не нужно.

**Примеры**

**входные данные**

2

5

0

2

**выходные данные**

2 4

**Задача 8. Делители числа.**

Выведите все натуральные делители числа *x* в порядке возрастания (включая 1 и само число).

**Входные данные**

Вводится натуральное число *x*

**Выходные данные**

Выведите все делители числа *x*

**Примеры**

**входные данные**

32

**выходные данные**

1 2 4 8 16 32

**Задача 9. Нули.**

**Входные данные**

Вводится число *N*, а затем *N* чисел.

**Выходные данные**

Подсчитайте и выведите, сколько среди данных *N* чисел нулей.

**Примеры**

**входные данные**

3

1

2

3

**выходные данные**

0

**Задача 10. Подсчет чисел.**

Подсчитайте, сколько среди данных *N* чисел нулей, положительных чисел, отрицательных чисел.

**Входные данные**

Вводится число *N*, а затем *N* целых чисел.

**Выходные данные**

Необходимо вывести сначала число нулей, затем число положительных и отрицательных чисел.

**Примеры**

**входные данные**

5

28

0

0

0

0

**выходные данные**

4 1 0

Раздел 3. Одномерные массивы.

**Задача 11. Целые точки отрезка**

Требуется написать программу, которая вычислит, сколько всего точек с целочисленными координатами принадлежат отрезку.

**Входные данные**

Даны четыре целых числа – координаты концов отрезка (*x*1, *y*1) и (*x*2, *y*2). Каждая из координат не превышает по абсолютной величине значения 1000.

**Выходные данные**

Требуется вывести количество точек отрезка, имеющих целочисленные координаты.

**Примеры**

**входные данные**

1 0 5 0

**выходные данные**

5

**входные данные**

-1 -2 2 4

**выходные данные**

4

**Задача 12. Отрезок 0.**

На клетчатой бумаге Петя нарисовал отрезок из точки с координатами (*a*,*b*) в точку с координатами (*c*,*d*). Через сколько клеток проходит этот отрезок (считается, что отрезок проходит через клетку, если он проходит через ее внутренность, если же он проходит только через вершину или по границе клетки, считается, что он не проходит через клетку).

**Входные данные**

Вводятся целые числа *a*, *b*, *c*, *d*. Числа по модулю не превышают 109.

**Выходные данные**

Выведите одно число — количество клеток, через которые проходит отрезок.

**Примеры**

**входные данные**

0 0 6 4

**выходные данные**

8

**входные данные**

3 3 -3 3

**выходные данные**

0

**Задача 13. Граница многоугольника.**

Многоугольник на плоскости задан целочисленными координатами своих *N* вершин в декартовой системе координат. Требуется найти количество точек с целочисленными координатами, лежащих на границе многоугольника. Стороны многоугольника друг с другом не соприкасаются (за исключением соседних - в вершинах) и не пересекаются.

Ограничения: 3 <= *N* <= 100 000, координаты вершин целые и по модулю не превосходят 1 000 000 000.

**Входные данные**

В первой строке содержится число *N*, в следующих *N* строках - пары чисел - координаты точек. Если соединить точки в данном порядке, а также соединить первую и последнюю точки, получится заданный многоугольник.

**Выходные данные**

Вывести одно число - количество точек с целочисленными координатами на границе многоугольника.

**Примеры**

**входные данные**

8

5 15

15 5

15 -5

5 -15

-5 -15

-15 -5

-15 5

-5 15

**выходные данные**

80

**Задача 14. Разрезание на квадраты.**

Полоска бумаги имеет размеры *A* × *B*. Каждый раз от нее отрезается квадрат максимального размера до тех пор, пока не получится квадрат. Сколько квадратов получится?

**Входные данные**

Программе даны числа *A* и *B* (1 ≤ *A*, *B* ≤ 109).

**Выходные данные**

Требуется вывести количество квадратов.

**Примеры**

**входные данные**

15 3

**выходные данные**

5

**входные данные**

12 8

**выходные данные**

3

**входные данные**

5 5

**выходные данные**

1

**Задача 15. Переливания.**

На досуге вы любите почитать сборники занимательных задач по математике. Недавно вы наткнулись в одном из таких сборников на следующую задачу:

*Есть бесконечный резервуар с водой и два пустых сосуда объёмом 5 и 12 литров. Можно наливать воду из резервуара в любой сосуд до его заполнения, переливать воду из —одного сосуда в другой до заполнения второго или опустошения первого (смотря что будет раньше) и выливать воду из сосуда на землю до полного опустошения сосуда. Как таким образом можно отмерить 3 литра?*

Вы решили написать программу, которая будет решать подобные задачи для произвольных объёмов сосудов.

**Входные данные**

Во входном файле находятся три целых числа — *V*1, *V*2 и *V* — объёмы двух сосудов и объем воды, который нужно отмерить. Гарантируется, что 1*V*1*V*232767 и 0*V*max(*V*1*V*2).

**Выходные данные**

В первую строку выходного файла выведите одно число — количество действий в вашем решении. Далее выведите соответствующее количество строк, описывающих действия в вашем решении. Для каждого действия выведите два числа:

* если это действие — переливание из одного сосуда в другой, то первое число должно быть номером сосуда, откуда надо переливать воду, а второе — номером сосуда, куда переливать;
* если это действие — набор воды из резервуара, то первое число должно быть нулём, а второе – номером сосуда, куда наливать;
* если это действие — выливание воды “на землю”, то первое число должно быть номером сосуда, а второе — нулём.

После выполнения всех операций хотя бы в одном сосуде должна находиться вода в объёме *V*.

Если существует несколько решений, то вы можете вывести любое. Ваше решение не обязано быть оптимальным, единственное ограничение — размер выходного файла не должен превосходить 3 Мб.

Если решений не существует, выведите одно число -1.

**Примеры**

**входные данные**

5 12 3

**выходные данные**

10

0 1

1 2

2 1

1 2

0 1

1 0

0 1

1 2

0 1

1 2

Раздел 4. Простые сортировки.

**Задача 16. Максимальный элемент в массиве.**

Напишите программу, которая находит значение максимального элемента массива.

**Входные данные**

В первой строке задается одно натуральное число *N*, не превосходящее 1000 – размер массива.

Во второй строке вводятся *N* чисел – элементы массива (целые числа, не превосходящие по модулю 1000).

**Выходные данные**

Вывести одно число – значение максимального элемента в массиве.

**Примеры**

**входные данные**

5

1 2 3 4 5

**выходные данные**

5

**Задача 17. Матрица.**

Задана матрица *K*, содержащая *n* строк и *m* столбцов. *Седловой точкой* этой матрицы назовем элемент, который одновременно является минимумом в своей строке и максимумом в своем столбце.

Найдите количество седловых точек заданной матрицы.

**Входные данные**

Первая строка содержит целые числа *n* и *m* (1 ≤ *n*, *m* ≤ 750). Далее следуют *n* строк по *m* чисел в каждой. *j*-ое число *i*-ой строки равно *k*ij. Все *k*ij по модулю не превосходят 1000.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

2 2

0 0

0 0

**выходные данные**

4

**входные данные**

2 2

1 2

3 4

**выходные данные**

1

**Задача 18. Двоичный поиск.**

Реализуйте алгоритм бинарного поиска.

**Входные данные**

В первой строке входных данных содержатся натуральные числа *N* и *K* (0*N**K*100000). Во второй строке задаются *N* элементов первого массива, отсортированного по возрастанию, а в третьей строке – *K* элементов второго массива. Элементы обоих массивов - целые числа, каждое из которых по модулю не превосходит 109

**Выходные данные**

Требуется для каждого из *K* чисел вывести в отдельную строку "YES", если это число встречается в первом массиве, и "NO" в противном случае.

**Примеры**

**входные данные**

10 5

1 2 3 4 5 6 7 8 9 10

-2 0 4 9 12

**выходные данные**

NO

NO

YES

YES

NO

**Задача 19. Левый и правый двоичный поиск.**

Дано два списка чисел, числа в первом списке упорядочены по неубыванию. Для каждого числа из второго списка определите номер первого и последнего появления этого числа в первом списке.

**Входные данные**

В первой строке входных данных записано два числа *N* и *M* (1*N**M*20000). Во второй строке записано *N* упорядоченных по неубыванию целых чисел — элементы первого списка. В третьей строке записаны *M* целых неотрицательных чисел - элементы второго списка. Все числа в списках - целые 32-битные знаковые.

**Выходные данные**

Программа должна вывести *M* строчек. Для каждого числа из второго списка нужно вывести номер его первого и последнего вхождения в первый список. Нумерация начинается с единицы. Если число не входит в первый список, нужно вывести одно число 0.

**Примеры**

**входные данные**

10 5

1 1 3 3 5 7 9 18 18 57

57 3 9 1 179

**выходные данные**

10 10

3 4

7 7

1 2

0

**Задача 20. Сортировка вставками.**

Требуется отсортировать массив по неубыванию методом "вставок".

**Входные данные**

В первой строке вводится одно натуральное число, не превосходящее 1000 – размер массива. Во второй строке задаются *N* чисел – элементы массива (целые числа, не превосходящие по модулю 1000).

**Выходные данные**

Вывести получившийся массив.

**Примеры**

**входные данные**

5

5 4 3 2 1

**выходные данные**

1 2 3 4 5

Раздел 5. Сортировка подсчетом.

**Задача 21. Сортировка подсчетом.**

Реализуйте алгоритм сортировки подсчетом для произвольных чисел, по модулю не превосходящих 10000.

**Входные данные**

На вход программе сначала подается значение *n* ≤ 100000 – количество элементов в массиве. В следующей строке входных данных расположены сами элементы массива – целые числа, по модулю не превосходящие 10000.

**Выходные данные**

Распечатайте отсортированный по неубыванию массив.

**Примеры**

**входные данные**

5

1 3 4 2 5

**выходные данные**

1 2 3 4 5

**Задача 22. Анаграммы.**

Слово называется анаграммой другого слова, если оно может быть получено перестановкой его символов.

**Входные данные**

Даны два слова на отдельных строках. Слова состоят из строчных латинских букв и цифр. Длины слов не превышают 255.

**Выходные данные**

Требуется вывести "YES" – если введенные слова являются анаграммами друг друга, "NO" – если нет.

**Примеры**

**входные данные**

sharm

marsh

**выходные данные**

YES

**входные данные**

ananas

nnaass

**выходные данные**

NO

#### Задача 23. Разброс

Дано *N* целых чисел, которые требуется отсортировать в порядке неубывания. В связи с нормами СЭС среди чисел не будет двух, разница между которыми превышает 107107.

**Входные данные**

Первая строка входного файла содержит целое число *N*. (1 <= *N* <= 100000), вторая строка – *N* целых чисел, не превышающих по модулю 2\*109. Никакие два не различаются более, чем на 107107.

**Выходные данные**

Выведите данные числа в порядке неубывания.

**Примеры**

**входные данные**

1

863961129

**выходные данные**

863961129

**входные данные**

5

1866455200 1866455199 1866455198 1866455197 1866455196

**выходные данные**

1866455196 1866455197 1866455198 1866455199 1866455200

#### Задача 24. Клавиатура

Всем известно, что со временем клавиатура изнашивается, и клавиши на ней начинают залипать. Конечно, некоторое время такую клавиатуру еще можно использовать, но для нажатий клавиш приходиться использовать большую силу.

При изготовлении клавиатуры изначально для каждой клавиши задается количество нажатий, которое она должна выдерживать. Если знать эти величины для используемой клавиатуры, то для определенной последовательности нажатых клавиш можно определить, какие клавиши в процессе их использования сломаются, а какие – нет.

Требуется написать программу, определяющую, какие клавиши сломаются в процессе заданного варианта эксплуатации клавиатуры.

**Входные данные**

Первая строка входного файла содержит целое число *n* (1 ≤ *n* ≤ 100) – количество клавиш на клавиатуре. Вторая строка содержит *n* целых чисел – *с*1, с2, … , *сn*, где *сi* (1 ≤ *сi* ≤ 100000) – количество нажатий, выдерживаемых *i*-ой клавишей. Третья строка содержит целое число *k* (1 ≤ *k* ≤ 100000) – общее количество нажатий клавиш, и последняя строка содержит *k* целых чисел *pj* (1 ≤ *pj* ≤ *n*) – последовательность нажатых клавиш.

**Выходные данные**

В выходной файл необходимо вывести ***n* строк**, содержащих информацию об исправности клавиш. Если *i*-ая клавиша сломалась, то *i*-ая строка должна содержать слово “**yes**” (без кавычек), если же клавиша работоспособна – слово “**no**”.

***Разбалловка для личной олимпиады***

Тест 1 — из условия. Оценивается в 0 баллов.

Тесты 2-21 — дополнительных ограничений нет. Группа тестов оценивается в 100 баллов.

Баллы начисляются за прохождение всех тестов группы и всех тестов предыдущих групп. При выставлении баллов за отдельные тесты каждый тест (кроме тестов из условия) оценивается в 5 баллов.

**Примеры**

**входные данные**

5

1 50 3 4 3

16

1 2 3 4 5 1 3 3 4 5 5 5 5 5 4 5

**выходные данные**

yes

no

no

no

yes

#### Задача 25. Палиндром\_0

Палиндром - это строка, которая читается одинаково как справа налево, так и слева направо.  
  
На вход программы поступает набор больших латинских букв (не обязательно различных). Разрешается переставлять буквы, а также удалять некоторые буквы. Требуется из данных букв по указанным правилам составить палиндром наибольшей длины, а если таких палиндромов несколько, то выбрать первый из них в алфавитном порядке.

**Входные данные**

В первой строке входных данных содержится число NN (1 <= NN <= 100000). Во второй строке задается последовательность из NN больших латинских букв (буквы записаны без пробелов).

**Выходные данные**

В единственной строке выходных данных выдайте искомый палиндром.

**Группы тестов**

25 баллов — (1 ≤ *N*≤ *10*).

25 баллов — (1 ≤ *N*≤ *1 000*).

50 баллов — полные ограничения.

**Примеры**

**входные данные**

3

AAB

**выходные данные**

ABA

**входные данные**

6

QAZQAZ

**выходные данные**

AQZZQA

**входные данные**

6

ABCDEF

**выходные данные**

A

Раздел 6. Двумерные массивы.

**Задача 26. Побочная диагональ.**

Дано число n, n  100. Создайте массив n×n и заполните его по следующему правилу:  
- числа на диагонали, идущей из правого верхнего в левый нижний угол, равны 1;  
- числа, стоящие выше этой диагонали, равны 0;  
- числа, стоящие ниже этой диагонали, равны 2.

**Входные данные**

Программа получает на вход число n.

**Выходные данные**

Необходимо вывести полученный массив. Числа разделяйте одним пробелом.

**Примеры**

**входные данные**

4

**выходные данные**

0 0 0 1

0 0 1 2

0 1 2 2

1 2 2 2

**Задача 27. Таблица умножения.**

Даны два числа n и m. Создайте двумерный массив A[n][m], заполните его таблицей умножения A[i][j]=i\*j и выведите на экран. При этом **нельзя использовать вложенные циклы**, все заполнение массива должно производиться одним циклом.

**Входные данные**

Программа получает на вход два числа n и m – количество строк и столбцов, соответственно.

**Выходные данные**

Программа должна вывести полученный массив. Числа разделяйте одним пробелом.

**Примеры**

**входные данные**

3 3

**выходные данные**

0 0 0

0 1 2

0 2 4

**Задача 28. Заполнение спиралью.**

Дано число n. Создайте массив A[2\*n+1][2\*n+1] и заполните его по спирали, начиная с числа 0 в центральной клетке A[n+1][n+1]. Спираль выходит вверх, далее закручивается против часовой стрелки.

**Входные данные**

Программа получает на вход одно число n.

**Выходные данные**

Программа должна вывести полученный массив, отводя на вывод каждого числа ровно 3 символа.

**Примеры**

**входные данные**

2

**выходные данные**

12 11 10 9 24

13 2 1 8 23

14 3 0 7 22

15 4 5 6 21

16 17 18 19 20

**входные данные**

8

2

**выходные данные**

1

**Задача 29. Заполнение диагоналями.**

Даны числа n и m. Создайте массив A[n][m] и заполните его, как показано на примере.

**Входные данные**

Программа получает на вход два числа n и m.

**Выходные данные**

Программа должна вывести полученный массив.

**Примеры**

**входные данные**

4 10

**выходные данные**

0 1 3 6 10 14 18 22 26 30

2 4 7 11 15 19 23 27 31 34

5 8 12 16 20 24 28 32 35 37

9 13 17 21 25 29 33 36 38 39

**Задача 30. Переворот.**

Дан массив *N* × *M*. Требуется повернуть его по часовой стрелке на 90 градусов.

**Входные данные**

На первой строке даны натуральные числа *N* и *M* (1 ≤ *N*, *M* ≤ 50). На следующих *N* строках записано по *M* неотрицательных чисел, не превышающих 109 – сам массив.

**Выходные данные**

Выведите повернутый массив в формате входных данных.

**Примеры**

**входные данные**

3 4

1 2 3 4

5 6 7 8

9 10 11 12

**выходные данные**

4 3

9 5 1

10 6 2

11 7 3

12 8 4

Раздел 7. Простые структуры данных. Списки, стеки, очереди.

**Задача 31. Стек с защитой от ошибок.**

Реализуйте структуру данных "стек". Напишите программу, содержащую описание стека и моделирующую работу стека, реализовав все указанные здесь методы. Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку. Возможные команды для программы:

**push n**

Добавить в стек число n (значение n задается после команды). Программа должна вывести ok.

**pop**

Удалить из стека последний элемент. Программа должна вывести его значение.

**back**

Программа должна вывести значение последнего элемента, не удаляя его из стека.

**size**

Программа должна вывести количество элементов в стеке.

**clear**

Программа должна очистить стек и вывести ok.

**exit**

Программа должна вывести bye и завершить работу.

Перед исполнением операций back и pop программа должна проверять, содержится ли в стеке хотя бы один элемент. Если во входных данных встречается операция back или pop, и при этом стек пуст, то программа должна вместо числового значения вывести строку error.

**Входные данные**

Вводятся команды управления стеком, по одной на строке

**Выходные данные**

Программа должна вывести протокол работы стека, по одному сообщению на строке

**Примеры**

**входные данные**

size

push 1

size

push 2

size

push 3

size

exit

**выходные данные**

0

ok

1

ok

2

ok

3

bye

**Задача 32. Очередь с защитой от ошибок.**

Реализуйте структуру данных "очередь". Напишите программу, содержащую описание очереди и моделирующую работу очереди, реализовав все указанные здесь методы.  Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку. Возможные команды для программы:

**push n**

Добавить в очередь число n (значение n задается после команды). Программа должна вывести ok.

**pop**

Удалить из очереди первый элемент. Программа должна вывести его значение.

**front**

Программа должна вывести значение первого элемента, не удаляя его из очереди.

**size**

Программа должна вывести количество элементов в очереди.

**clear**

Программа должна очистить очередь и вывести ok.

**exit**

Программа должна вывести bye и завершить работу.

Перед исполнением операций front и pop программа должна проверять, содержится ли в очереди хотя бы один элемент. Если во входных данных встречается операция front или pop, и при этом очередь пуста, то программа должна вместо числового значения вывести строку error.

**Входные данные**

Вводятся команды управления очередью, по одной на строке

**Выходные данные**

Требуется вывести протокол работы очереди, по одному сообщению на строке

**Примеры**

**входные данные**

push 1

front

exit

**выходные данные**

ok

1

bye

**входные данные**

size

push 1

size

push 2

size

push 3

size

exit

**выходные данные**

0

ok

1

ok

2

ok

3

bye

**Задача 33. Простой дек.**

Реализуйте структуру данных "дек".  Напишите программу, содержащую описание дека и моделирующую работу дека, реализовав все указанные здесь методы. Программа считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения каждой команды программа должна вывести одну строчку. Возможные команды для программы:

**push\_front**

Добавить (положить) в начало дека новый элемент. Программа должна вывести ok.

**push\_back**

Добавить (положить) в конец дека новый элемент. Программа должна вывести ok.

**pop\_front**

Извлечь из дека первый элемент. Программа должна вывести его значение.

**pop\_back**

Извлечь из дека последний элемент. Программа должна вывести его значение.

**front**

Узнать значение первого элемента (не удаляя его). Программа должна вывести его значение.

**back**

Узнать значение последнего элемента (не удаляя его). Программа должна вывести его значение.

**size**

Вывести количество элементов в деке.

**clear**

Очистить дек (удалить из него все элементы) и вывести ok.

**exit**

Программа должна вывести bye и завершить работу.

Гарантируется, что количество элементов в деке в любой момент не превосходит 100. Все операции pop\_front, pop\_back, front, back всегда корректны.

**Входные данные**

Вводятся команды управления деком, по одной на строке.

**Выходные данные**

Требуется вывести протокол работы дека, по одному сообщению на строке.

**Примеры**

**входные данные**

push\_back 1

back

exit

**выходные данные**

ok

1

bye

**входные данные**

size

push\_back 1

size

push\_back 2

size

push\_front 3

size

exit

**выходные данные**

0

ok

1

ok

2

ok

3

bye

**Задача 34. Постфиксная запись.**

В постфиксной записи (или обратной польской записи) операция записывается после двух операндов. Например, сумма двух чисел A и B записывается как A B +. Запись B C + D \* обозначает привычное нам (B + C) \* D, а запись A B C + D \* + означает A + (B + C) \* D. Достоинство постфиксной записи в том, что она не требует скобок и дополнительных соглашений о приоритете операторов для своего чтения.

**Входные данные**

В единственной строке записано выражение в постфиксной записи, содержащее цифры и операции +, -, \*. Числа и операции разделяются пробелами. В конце строки может быть произвольное количество пробелов.

**Выходные данные**

Необходимо вывести значение записанного выражения.

**Примеры**

**входные данные**

8 9 + 1 7 - \*

**выходные данные**

-102

**Задача 35. контейнеры.**

На складе хранятся контейнеры с товарами *N* различных видов. Все контейнеры составлены в *N* стопок. В каждой стопке могут находиться контейнеры с товарами любых видов (стопка может быть изначально пустой).

Автопогрузчик может взять верхний контейнер из любой стопки и поставить его сверху в любую стопку. Необходимо расставить все контейнеры с товаром первого вида в первую стопку, второго вида – во вторую стопку и т. д.

Программа должна вывести последовательность действий автопогрузчика или сообщение о том, что задача решения не имеет.

***Формат входных данных***

В первой строке входных данных записано одно натуральное число *N*, не превосходящее 500. В следующих *N* строках описаны стопки контейнеров: сначала записано число *ki* – количество контейнеров в стопке, а затем *ki* чисел – виды товара в контейнерах в данной стопке, снизу вверх. В каждой стопке вначале не более 500 контейнеров (в процессе переноса контейнеров это ограничение может быть нарушено).

***Формат выходных данных***

Программа должна вывести описание действий автопогрузчика: для каждого действия напечатать два числа – из какой стопки брать контейнер и в какую стопку класть. (Обратите внимание, что минимизировать количество операций автопогрузчика не требуется.) Если задача не имеет решения, необходимо вывести одно число 0. Если контейнеры изначально правильно размещены по стопкам, то выводить ничего не нужно.

***Пример***

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
| 3 4 1 2 3 2 0 0 | 1 2 1 3 1 2 |

Раздел 8. Работа со строками.

**Задача 36. Делаем срезы.**

Выполните разбор строки, согласно приведенным правилам.

**Входные данные**

Дана строка.

**Выходные данные**

Сначала выведите третий символ этой строки.

Во второй строке выведите предпоследний символ этой строки.

В третьей строке выведите первые пять символов этой строки.

В четвертой строке выведите всю строку, кроме последних двух символов.

В пятой строке выведите все символы с четными индексами (считая, что индексация начинается с 0, поэтому символы выводятся начиная с первого).

В шестой строке выведите все символы с нечетными индексами, то есть начиная со второго символа строки.

В седьмой строке выведите все символы в обратном порядке.

В восьмой строке выведите все символы строки через один в обратном порядке, начиная с последнего.

В девятой строке выведите длину данной строки.

**Примеры**

**входные данные**

Abrakadabra

**выходные данные**

r

r

Abrak

Abrakadab

Arkdba

baaar

arbadakarbA

abdkrA

11

**Задача 37. Переставить два слова.**

Дана строка, состоящая ровно из двух слов, разделенных пробелом. Переставьте эти слова местами. Результат запишите в строку и выведите получившуюся строку.

При решении этой задачи нельзя пользоваться циклами и инструкцией if.

**Входные данные**

Вводится строка.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

Hello, world!

**выходные данные**

world! Hello,

**Задача 38. Замена подсроки.**

Дана строка. Замените в этой строке все цифры 1 на слово one.

**Входные данные**

Вводится строка.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

1+1=2

**выходные данные**

one+one=2

**Задача 39. Замена внутри фрагмента.**

Дана строка. Замените в этой строке все появления буквы h на букву H, кроме первого и последнего вхождения.

**Входные данные**

Вводится строка.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

In the hole in the ground there lived a hobbit

**выходные данные**

In the Hole in tHe ground tHere lived a hobbit

**Задача 40. Удалить каждый третий символ.**

Дана строка. Удалите из нее все символы, чьи индексы делятся на 3.

**Входные данные**

Вводится строка.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

Python

**выходные данные**

yton

Раздел 9. Сложные сортировки.

**Задача 41. Переставить в обратном порядке.**

Переставьте элементы данного списка в обратном порядке, затем выведите элементы полученного списка.

Эта задача отличается от предыдущей тем, что вам нужно изменить значения элементов самого списка, поменяв местами A[0] c A[n-1], A[1] с A[n-2], а затем вывести элементы списка подряд.

**Входные данные**

Вводится список чисел. Все числа списка находятся на одной строке.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

1 2 3 4 5

**выходные данные**

5 4 3 2 1

**Задача 42. Игра со спичками.**

На столе лежит кучка из *N* спичек. Двое играют в такую игру. За один ход разрешается взять из кучки одну, две или три спички, так чтобы оставшееся количество спичек не было простым числом (Например, можно оставить в кучке 1 или 4 спички, но нельзя оставить 2 или 3). Выигрывает тот, кто забирает последнюю спичку. Требуется определить, кто из игроков имеет выигрышную стратегию.

**Входные данные**

Вводится одно число *N* (1<=*N*<=10000).

**Выходные данные**

Выведите число 1, если выигрышую стратегию имеет начинающий игрок, или число 2, если выигрышную стратегию имеет второй игрок.

**Примеры**

**входные данные**

1

**выходные данные**

1

**Задача 43. Вывести четные элементы.**

Дан массив, состоящий из целых чисел. Напишите программу, которая выводит те элементы массива, которые являются чётными числами.

**Входные данные**

Сначала задано число *N* — количество элементов в массиве (1*N*100). Далее через пробел записаны *N* чисел — элементы массива. Массив состоит из целых чисел.

**Выходные данные**

Необходимо вывести все четные элементы массива (то есть те элементы, которые являются четными числами).

**Примеры**

**входные данные**

5

1 2 3 4 5

**выходные данные**

2 4

**Задача 44. Циклический сдвиг вправо.**

Напишите программу, которая циклически сдвигает элементы массива вправо (например, если элементы нумеруются, начиная с нуля, то 0-й элемент становится 1-м, 1-й становится 2-м, ..., последний становится 0-м, то есть массив {3, 5, 7, 9} превращается в массив {9, 3, 5, 7}).

**Входные данные**

Сначала задано число *N* — количество элементов в массиве (1*N*35). Далее через пробел записаны *N* чисел — элементы массива. Массив состоит из целых чисел.

**Выходные данные**

Необходимо вывести массив, полученный после сдвига элементов.

**Примеры**

**входные данные**

6

4 5 3 4 2 3

**выходные данные**

3 4 5 3 4 2

**Задача 45. Шарики.**

В одной компьютерной игре игрок выставляет в линию шарики разных цветов. Когда образуется непрерывная цепочка из трех и более шариков одного цвета, она удаляется из линии. Все шарики при этом сдвигаются друг к другу, и ситуация может повториться.

Напишите программу, которая по данной ситуации определяет, сколько шариков будет "уничтожено". Естественно, непрерывных цепочек из трех и более одноцветных шаров в начальный момент может быть не более одной.

**Входные данные**

Сначала вводится количество шариков в цепочке (не более 1000) и цвета шариков (от 0 до 9, каждому цвету соответствует свое целое число).

**Выходные данные**

Требуется вывести количество шариков, которое будет "уничтожено".

**Примеры**

**входные данные**

5 1 3 3 3 2

**выходные данные**

3

**XII. Список использованных источников и литературы**

**Основная литература:**

1. Гаврилов, А.В. Программирование на Java. Конспект лекций [Электронный ресурс] : учебное пособие / А.В. Гаврилов, С.В. Клименков, Е.А. Цопа. — Электрон. дан. — Санкт-Петербург: НИУ ИТМО, 2010. — 130 с. — Режим доступа: https://e.lanbook.com/book/43547

2. Мейер Б. Инструменты, алгоритмы и структуры данных. М.: ИНТУИТ, 2016. – 542 с. [Электронный ресурс] http://www.iprbookshop.ru/73680.html

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

**Дополнительная литература:**

4. Дубаков, А.А. Введение в объектно-ориентированное программирование на Java [Электронный ресурс]: учебное пособие / А.А. Дубаков. — Электрон. дан. — Санкт-Петербург: НИУ ИТМО, 2016. — 250 с. — Режим доступа: https://e.lanbook.com/book/110468

5. 8. Алексеев В. Е., Таланов В. А. Графы и алгоритмы. Структуры данных. Модели вычислений. Изд.: Интернет- университет ИТ, 2006. – 320 с.

6. 9. Ершов Ю. Математическая логика. Изд.: Лань, 2005, -336 с.

7. 10. Стивен С. Скиена. Алгоритмы. Руководство по разработке. Изд.: Springer, 2011. – 720 с.

**ПРИЛОЖЕНИЕ**

Тест. Входной контроль знаний по программе «Разработчик JAVA. Продолжающий уровень»

**Входное тестирование**

**Задача 1. Упорядочить 3 числа.**

Дано три числа, записанный в отдельных строках. Упорядочите их в порядке неубывания. Программа должна считывать три числа a, b, c, затем программа должна менять их значения так, чтобы стали выполнены условия a <= b <= c, затем программа выводит тройку a, b, c.

**Входные данные**

Вводятся три числа, каждое записано в отдельной строке.

**Выходные данные**

Выведите ответ на задачу.

**Примеры**

**входные данные**

1

2

1

**выходные данные**

1 1 2

**Задача 2. Четные числа.**

**Входные данные**

Вводятся целые числа *a* и *b*. Гарантируется, что *a* не превосходит *b*

**Выходные данные**

Выведите (через пробел) все четные числа от *a* до *b* (включительно).

**Примеры**

**входные данные**

2

5

**выходные данные**

2 4

**Задача 3. Остаток.**

**Входные данные**

Вводятся 4 числа: *a*, *b*, *c* и *d*.

**Выходные данные**

Выведите все числа на отрезке от *a* до *b*, дающие остаток *c* при делении на *d*. Если таких чисел не существует, то ничего выводить не нужно.

**Примеры**

**входные данные**

2

5

0

2

**выходные данные**

2 4

**Задача 4. Делители числа.**

Выведите все натуральные делители числа *x* в порядке возрастания (включая 1 и само число).

**Входные данные**

Вводится натуральное число *x*

**Выходные данные**

Выведите все делители числа *x*

**Примеры**

**входные данные**

32

**выходные данные**

1 2 4 8 16 32

**Задача 5. Нули.**

**Входные данные**

Вводится число *N*, а затем *N* чисел.

**Выходные данные**

Подсчитайте и выведите, сколько среди данных *N* чисел нулей.

**Примеры**

**входные данные**

3

1

2

3

**выходные данные**

0

**Задача 6. Подсчет чисел.**

Подсчитайте, сколько среди данных *N* чисел нулей, положительных чисел, отрицательных чисел.

**Входные данные**

Вводится число *N*, а затем *N* целых чисел.

**Выходные данные**

Необходимо вывести сначала число нулей, затем число положительных и отрицательных чисел.

**Примеры**

**входные данные**

5

28

0

0

0

0

**выходные данные**

4 1 0

Тест. Итоговый контроль знаний по программе «Разработчик JAVA. Продолжающий уровень»

**Итоговое тестирование**

Задача 1. Дано число N. Определите, сколькими способами можно расставить на доске N×N N ферзей, не бьющих друг друга.

Входные данные

Задано единственное число N. (N ≤ 10)

Выходные данные

Необходимо вывести количество способов, которыми можно расставить на доске N×N N ферзей, не бьющих друг друга.

Примеры

входные данные

8

выходные данные

92

Задача 2. На прямой задано некоторое множество отрезков с целочисленными координатами концов [Li, Ri]. Выберите среди данного множества подмножество отрезков, целиком покрывающее отрезок [0, M], (M — натуральное число), содержащее наименьшее число отрезков.

Входные данные

В первой строке указана константа M (1M5000). В каждой последующей строке записана пара чисел Li и Ri (LiRi50000), задающая координаты левого и правого концов отрезков. Список завершается парой нулей. Общее число отрезков не превышает 100 000.

Выходные данные

В первой строке выходного файла выведите минимальное число отрезков, необходимое для покрытия отрезка [0; M]. Далее выведите список покрывающего подмножества, упорядоченный по возрастанию координат левых концов отрезков. Список отрезков выводится в том же формате, что и во входe. Завершающие два нуля выводить не нужно. Если покрытие отрезка [0, M] исходным множеством отрезков [Li, Ri] невозможно, то следует вывести единственную фразу “No solution”.

Примеры

входные данные

1

-1 0

-5 -3

2 5

0 0

выходные данные

No solution

входные данные

1

-1 0

0 1

0 0

выходные данные

1

0 1

Задача 3. Дан массив. Надо научиться обрабатывать два типа запросов.

\* 1 L R - перевернуть отрезок [LR]

\* 2 L R - найти минимум на отрезке [LR]

Входные данные

Первая строка файла содержит два числа n, m. (1nm105) Во второй строке находится n чисел ai (1ai109) — исходный массив. Остальные m строк содержат запросы, в формате описанном в условии. Для чисел L, R выполняется ограничение (1LRn).

Выходные данные

На каждый запрос типа 2, во входной файл выведите ответ на него, в отдельной строке.

Примеры

входные данные

10 7

5 3 2 3 12 6 7 5 10 12

2 4 9

1 4 6

2 1 8

1 1 8

1 8 9

2 1 7

2 3 6

выходные данные

3

2

2

2

Задача 4. Дан граф из n вершин, раскрасьте его в минимально возможное число цветов так, чтобы никакие две вершины, соединенные ребром, не были одного цвета.

Входные данные

В первой строке содержится число t - количество тестовых примеров (1t5).

Далее содержится t тестовых случаев, заданных в следующем формате:

В первой строке записаны числа n и m - количество вершин и ребер соответственно (1n17, 0m2n(n−1)).

Затем идет m строк, в которых содержится по два числа vi ui, что означает, что вершины vi и ui соеденены ребром (1viuinvi=ui).

Гарантируется, что все ребра в каждом тестовом случае различны.

Выходные данные

Для каждого тестового случая в первой строке выведите минимальное число цветов k.

Во второй строке выведите n чисел ai - цвета вершин (1aik).

Примеры

входные данные

3

3 3

1 2

2 3

3 1

5 3

2 1

3 1

4 2

6 7

1 2

1 5

2 5

2 3

2 4

5 6

5 4

выходные данные

3

3 2 1

2

1 2 2 1 1

3

1 3 1 1 2 1

Задача 5. Представьте данное число n в виде суммы двух кубов.

Входные данные

Программа получает на вход одно натуральное число n(n <= 1028).

Выходные данные

Программа должна вывести 2 целых неотрицательных числа, сумма кубов которых равна n. Если это невозможно, выведите строку impossible.

Примеры

входные данные

2

выходные данные

1 1

входные данные

3

выходные данные

impossible

Задача 6. Задана матрица K, содержащая n строк и m столбцов. Седловой точкой этой матрицы назовем элемент, который одновременно является минимумом в своей строке и максимумом в своем столбце.

Найдите количество седловых точек заданной матрицы.

Входные данные

Первая строка содержит целые числа n и m (1 ≤ n, m ≤ 750). Далее следуют n строк по m чисел в каждой. j-ое число i-ой строки равно kij. Все kij по модулю не превосходят 1000.

Выходные данные

Выведите ответ на задачу.

Примеры

входные данные

2 2

0 0

0 0

выходные данные

4

входные данные

2 2

1 2

3 4

выходные данные

1