**ПРИМЕРНАЯ ДОПОЛНИТЕЛЬНАЯ ОБЩЕРАЗВИВАЮЩАЯ ПРОГРАММА**

**«Программирование на языке высокого уровня»**

Направленность: техническая
Уровень реализации программы: базовый
Возраст: 15-17 лет
Срок реализации: 1 год (72 часа)

Москва 2019г.

# РЕЦЕНЗИЯ

**на примерную дополнительную общеразвивающую программу**

**«Программирование на языке высокого уровня»**

**для обучающихся 15-17 лет.**

Представленная на рецензию примерная дополнительная общеразвивающая программа актуальна и ориентирована на формирование современных компетенций в области программирования на языке высокого уровня, применяемого для решения различных задач в разных областях: от разработки мобильных приложений до машинного обучения.

Основной целью примерной программы является обеспечение целостного компетентного обучения базовым навыкам программирования на современном профессиональном языке программирования.

В структуре рецензируемой примерной программы присутствуют: пояснительная записка, описание образовательной новизны программы, общая характеристика курса, описание форм организаций учебных занятий, ожидаемые результаты и способы определения их результативности, учебно-тематическое планирование, описание форм аттестации и оценочные материалы, организационно – педагогические условия реализации программы, перечень информационных ресурсов.

Описанные в примерной программе методические подходы, выбранное предметное содержание и материально-техническое оснащение соответствуют заявленным в примерной программе цели и задачам, а также возрастным особенностям обучающихся.

Таким образом, рецензируемая примерная дополнительная общеразвивающая программа «Программирование на языке высокого уровня» соответствует требованиям, предъявляемым к документам данного типа.

**И.о. декана инженерной школы (факультета)** **И.Н. Вольнов**

Подпись И.Н. Вольнова заверяю
ЗАМЕСТИТЕЛЬ НАЧАЛЬНИКА
ОТДЕЛА КАДРОВ
Пономарева А.А.

**Оглавление**

## 1. Пояснительная записка

**Направленность программы** – техническая.
**Уровень программы** – базовый.
**Возраст обучающихся**: от 15 лет до 17 лет.
**Срок реализации программы:** 1 год, 72 часа.

**Актуальность** программы определяется тем, что в любой профессиональной области сегодня требуется вычислительное и алгоритмическое мышление, что позволяет решать сложные нестандартные задачи не только в IT-области, но и в повседневной жизни.

Изучение основных принципов программирования невозможно без регулярной практики написания программ на каком-либо языке. В представленной программе большое внимание уделяется практической работе, самостоятельному написанию кода. Для реализации данной программы рекомендуются на выбор один из следующих языков программирования: Python, C, C++, C#, Java, Swift, Scala. Данные языки программирования являются востребованными, они подходят для знакомства с различными современными парадигмами программирования и активно применяются в самых разных областях от разработки мобильных приложений до машинного обучения.

Знания и умения, приобретённые в результате освоения курса, могут быть использованы обучающимися при решении задач по физике, химии, биологии и другим наукам, а также они являются фундаментом для дальнейшего совершенствования мастерства программирования.

Курс «Программирование на языке высокого уровня» рассчитан на 72 учебных часа и предназначен для учеников 10 и/или 11 классов, имеющих начальный уровень подготовки в области алгоритмизации.

## 2. Новизна образовательной программы

Новизна заключается в практической значимости курса, что способствует более успешному овладению знаниями и умениями по направлению «Программирование» через развитие самостоятельности обучающихся и оптимизацию средств и методов обучения.

Программа курса:

- имеет практическую направленность с ориентацией на реальные потребности, соответствующие возрасту учащихся;
- охватывает вопросы практического использования полученных знаний при решении задач из различных областей знаний;
- ориентирована на существующий парк вычислительной техники;
- допускает возможность варьирования в зависимости от уровня подготовки и интеллектуального уровня учащихся (как группового, так и индивидуального);
- предусматривает возможность индивидуальной работы с учащимися.

**Педагогическая целесообразность** состоит в том, что учащиеся могут подготовиться к программно-технической деятельности с дальнейшим самоопределением и развитием в IT-области.

**Цель:** обеспечить целостное компетентное обучение базовым навыкам программирования на современном профессиональном языке программирования.

**Задачи:**

- закрепить и расширить знания учащихся по алгоритмизации и программированию;
- привить навыки решения задач по программированию;
- воспитывать аккуратность, самостоятельность, умение работать в команде, информационную и коммуникационную культуры;
- воспитывать усидчивость и методичность при реализации проекта.

**Отличительные особенности программы:** данная программа отличается от уже существующих тем, что в ней нет описания конкретного языка программирования. Программа, как шаблон, отражает ключевые аспекты любого языка программирования, что позволяет на ее основе обучать учащихся любым программным средам.

## 3. Общая характеристика курса «Основы программирования на языке высокого уровня»

### 3.1. Основные разделы программы

***Раздел 1. Введение в программирование***

Основные понятия программирования: исполнитель, система команд, алгоритм, программа, среда разработки, интерпретатор, код программы и редактор кода. Технология разработки программы. Жизненный цикл программы. Понятие о языке высокого уровня. Структура программы, переменные и константы, работа с числовыми переменными, арифметические операторы.

***Раздел 2. Базовые конструкции языка программирования***

Ввод-вывод в программе, основные управляющие конструкции алгоритмов с ветвлением, условный оператор. Программы с использованием операторов ввода-вывода, условного оператора.

Основные управляющие конструкции циклического алгоритма. Программы с использованием циклов.

***Раздел 3. Решение прикладных задач***

Работа со списками, строками, множествами и кортежами. Понятие итератора. Подпрограммы, процедуры, функции.

Массивы. Словари. Модули. Подключение и использование модулей библиотек. Модульный принцип компоновки программы. Работа в стандартной библиотеке. Репозитории различных пакетов. Работа с внешними библиотеками.

***Раздел 4. Выполнение индивидуальной или совместной работы.***

Каждый ученик или группа из двух - трех учащихся должны выполнить проект на заданную тему (или по выбору учащихся), в ходе работы над которым демонстрируется вся сумма знаний и практических навыков, полученных в ходе обучения.

Проектная работа разбивается на следующие этапы:

- проект на бумаге; полное описание - техническое задание на проект.
- компьютерная реализация проекта; выполняется учениками на нескольких занятиях; педагог контролирует процесс выполнения работы, отвечает на возникающие вопросы, **консультирует**.

**Защита проектов.** Зачётное занятие: защита индивидуальной или совместной работы. Выполненная работа демонстрируется всей группе; автор (группа авторов) представляет проект; группа обсуждает представленный проект; автор (авторы) отвечает на вопросы.

## 3.2. Формы организации учебных занятий

*Форма и режим занятий:* Занятия проводятся 1 раз в неделю по 2 часа в групповой форме, включают в себя 45 минут учебного времени и 15 мин обязательный перерыв.

Единицей учебного процесса является блок уроков (раздел). Каждый раздел охватывает отдельную информационную технологию или её часть. Внутри раздела разбивка по времени изучения производится учителем самостоятельно, но с учётом рекомендованного учебно-тематического плана.

Закрепление знаний проводится с помощью практики отработки умений самостоятельно решать поставленные задачи, соответствующих минимальному уровню планируемых результатов обучения.

Задания выполняются на компьютере с использованием интегрированной среды разработки. При этом ученики не только формируют новые теоретические и практические знания, но и приобретают новые технологические навыки.

Для самостоятельной работы используются разные по уровню сложности тренировочные упражнения, которые носят репродуктивный и творческий характер. Количество таких упражнений в работе может варьироваться.

В ходе обучения проводится промежуточное тестирование по темам для определения уровня знаний учащихся.

Выполнение тренировочных упражнений и тестирование способствует активизации учебно-познавательной деятельности и ведёт к закреплению знаний, а также служит индикатором успешности образовательного процесса.

*Формы проведения занятий*

***Разъяснение теоретического материала.*** Может проводиться в виде представления презентации или видеоурока, содержащего необходимый учебный материал. Презентация (видеоурок) может просматриваться совместно с помощью проектора или открываться как сетевой ресурс каждым учащимся на своем компьютере и просматриваться в удобном для него темпе (демонстрационный или наглядный метод).

***Практическое освоение нового материала.*** Выполнение тренировочных упражнений на каждом занятии на компьютере под контролем педагога

***Индивидуальная работа по закреплению пройденного материала.*** Индивидуальное задание выдается каждому учащемуся. (Возможен вариант работы в парах).

***Индивидуальная работа с учащимися.*** Педагог дает индивидуальное задание повышенной сложности или помогает учащемуся поставить задачу и реализовать свой творческий замысел.

***Тестирование.*** Выполняется с целью закрепления изученного материала.

***Итоговая работа.*** Завершает изучение всего материала. Чтобы продемонстрировать всю сумму знаний и практических навыков, каждый ученик или группа из двух - трех учащихся должны выполнить проект на заданную тему или по выбору учащихся.

*Формы и методы контроля:*

- тестирование;
- выполнение тренировочных упражнений;
- выполнение итогового проекта

*Характеристика учебного процесса:*

- при изучении курса используются практические самостоятельные работы;
- курс обучения заканчивается выполнением и защитой индивидуальной или совместной итоговой работы.

## Ожидаемые результаты и способы определения их результативности

| Будут знать | Будут уметь | Форма подведения итогов |
|---|---|---|
| Правила по технике безопасности. | Соблюдать правила техники безопасности на занятиях | По окончании курса учащиеся создают индивидуально или в команде (не более 3 человек) итоговую работу, включающую в себя все ранее изученные аспекты изученного языка программирования. |
| Основные понятия программирования, Технологию разработки программы. | Работать в среде программирования, запускать и отлаживать программы | |
| Базовые конструкции языка программирования | Составлять запускать и отлаживать программы на изучаемом языке программирования, используя линейные, ветвящиеся и циклические алгоритмы, определять и использовать вспомогательные алгоритмы. | |
| Методы решения различных по степени сложности задач по программированию. | Кодировать на изучаемом языке программирования различные по степени сложности задачи. | |
| Технологии построения простых и сложных алгоритмов на предложенном языке программирования | Использовать среду реализации изучаемого языка программирования для решения конкретной задачи | |

Для **подведения итогов** реализации программы предусмотрена аттестация в форме выполнения и защиты итоговой индивидуальной или совместной работы.

**Учебно-тематический план**

| № | Название раздела, темы | Всего | В том числе | | Форма аттестации (контроля) |
|---|---|---|---|---|---|
| | | | Теория | Практика | |
| **1** | **Раздел 1. Введение в программирование.** | **8** | **2,5** | **5,5** | |
| 1.1 | Основные понятия программирования. Понятие о языке высокого уровня. | 2 | 2 | - | Тестирование |
| 1.2 | Технология разработки программы. | 6 | 0,5 | 5,5 | Практическая работа |
| **2** | **Раздел 2. Базовые конструкции языка программирования.** | **16** | **3** | **13** | |
| 2.1 | Основные управляющие конструкции в программировании. | 16 | 3 | 13 | ТестированиеПрактическая работа |
| **3** | **Раздел 3. Решение прикладных задач.** | **34** | **8** | **26** | |
| 3.1 | Работа со списками, строками, множествами и кортежами. Понятие итератора. | 8 | 2 | 6 | Практическая работа |
| 3.2 | Подпрограммы, процедуры, функции. | 6 | 1 | 5 | Практическая работа |
| 3.3 | Массивы. Словари. Модули. | 8 | 2 | 6 | Практическая работа |
| 3.4 | Подключение и использование модулей библиотек. | 4 | 1 | 3 | Практическая работа |
| 3.5 | Стандартная библиотека | 4 | 1 | 3 | Практическая работа |
| 3.6 | Репозитории. Внешние библиотеки. | 4 | 1 | 3 | Практическая работа |
| **4** | **Раздел 4. Выполнение индивидуального или совместного итогового проекта.** | **10** | **1,5** | **8,5** | |
| **5** | **Защита итогового проекта** | **4** | **-** | **4** | Защита проекта |
| | Итого: | **72** | **15** | **57** | |

## 4. Содержание программы

**Раздел 1. Введение в программирование.**

**Тема 1. 1** Общая информация. Правила по технике безопасности при работе с оборудованием в классе. Термины и определения.

*Теория (2 ч.)* Знакомство с учащимися. Правила поведения и правила по технике безопасности на занятиях (Инструктаж). Основные термины и определения в программировании. Знакомство с историей языков программирования. Знакомство с классификациями языков программирования по различным характеристикам (низкого, высокого и сверхвысокого уровней, интерпретаторы и компиляторы, процедурные, обектно-ориентированные и т.д.) и их применением. Знакомство учащихся с тем, что язык программирования предопределяет профессиональную деятельность.

Основные понятия программирования: исполнитель, система команд, алгоритм, программа, среда разработки, интерпретатор, код программы и редактор кода.

**Тема 1.2.** Технология разработки программы.

*Теория (0,5 ч.)* Структура программы, переменные и константы, работа с числовыми переменными, арифметические операторы.

*Практика (5,5 ч.)* Знакомство с интегрированной средой программирования, исполнение кода. Обучающиеся разрабатывают первые программы, а также анализируют, на какие функциональные блоки может быть разбита программа, и определяют работоспособность программы.

**Форма контроля по темам Раздела 1: тестирование.**

Форма контроля подразумевает тестирование учащихся по вопросам пройденных тем.

**Раздел 2. Базовые конструкции языка программирования.**

**Тема 2.1.** Основные управляющие конструкции в программировании.

*Теория (3 ч.)* Ввод-вывод в программе, основные управляющие конструкции алгоритмов с ветвлением. Простейшие программы с использованием операторов ввода-вывода, условного оператора. Основные управляющие конструкции циклического алгоритма.

*Практика (13 ч.)* Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Форма контроля по темам раздела 2: практическая работа.**

Форма контроля по разделу представляет собой тестирование, а также демонстрацию работы программ для тренировочных упражнений.

**Раздел 3. Решение прикладных задач.**

**Тема 3.1** Работа со списками, строками, множествами, кортежами. Понятие итератора.

*Теория (2 ч.)* Списки, строки, множества и кортежи. Понятие итератора.

*Практика (6 ч.)* Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Тема 3.2.** Подпрограммы, процедуры, функции.

*Теория (1 ч.)* Подпрограммы, процедуры, функции.

*Практика (7 ч.)* Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Тема 3.3.** Массивы. Словари. Модули.

*Теория (2 ч.)* Обработка массивов. Словари. Модульный принцип компоновки программы.

*Практика (6ч.)* Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Тема 3.4.** Подключение и использование модулей библиотек.

*Теория (1 ч.)*

*Практика (3 ч.)* Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Тема 3.5.** Стандартная библиотека.

*Теория (1 ч.)*. Работа в стандартной библиотеке.

*Практика (3 ч.)*. Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Тема 3.6.** Репозитории. Внешние библиотеки.

*Теория (1 ч.)* Репозитории различных пакетов. Работа с внешними библиотеками.

*Практика (3 ч.)*. Разработка кода для тренировочных упражнений. Набор, отладка и запуск программ.

**Форма контроля по темам раздела 3: практическая работа.**

Форма контроля представляет собой демонстрацию работы программ для тренировочных упражнений.

**Раздел 4. Выполнение индивидуального или совместного итогового проекта.**

*Теория (1 ч.)* Выбор тематики итогового проекта, разработка индивидуальных вариантов реализации проекта. Разработка технического задания на проект.

*Практика (5 ч.)* Самостоятельная практическая работа над созданием итогового проекта. Отладка, обработка и оптимизация программных кодов.

**Раздел 5. Защита итогового проекта.**

*Практика (4 ч.)* Демонстрация учащимися выполненных итоговых проектов. Обсуждение и оценивание итоговых проектов.

## 5. Формы аттестации и оценочные материалы

Формы аттестации:

- в течение занятий – экспресс-опросы учащихся в форме «вопрос-ответ», тестирование
- выполнение тренировочных упражнений
- по окончании курса – выполнение итогового проекта.

Защита итогового проекта проходит в форме представления обучающимся технического задания на проект, работающего кода, ответов на вопросы преподавателя. Обсуждения с учащимися достоинств и недостатков проекта.

**Критерии оценивания итогового проекта:**

– самостоятельность выполнения,

– законченность работы,

– соответствие выбранной тематике,

– использование при работе над проектом основных аспектов языка программирования, изученных в ходе обучения.

**Примеры тренировочных упражнений**

1. Реализовать приложение, которое для введенной последовательности из N целых чисел находит максимальное из чисел, минимальное из чисел, вычисляет среднее арифметическое, сумму введенных чисел, сортирует числа в порядке возрастания, убывания.
2. Реализовать приложение, которое для введенного текста осуществляет поиск подстроки в строке, определение количества слов, количества символов с учетом пробелов, без учета пробелов, количества гласных букв, поиск слов с максимальной/минимальной длиной, работу со спецсимволами.
3. Реализовать библиотеку классов, представляющих геометрические фигуры - окружность, квадрат, треугольник. Реализовать методы вычисления основных параметров (площади, периметра, длины ребер, др.). В качестве

основы библиотеки создать интерфейс (или абстрактный класс) GeomFigure. Предусмотреть конструкторы с параметрами.

4. Для упражнений 1-3 дополнить реализованные ранее приложения обработкой исключений. Реализовать библиотеку классов - абстрактную модель организационной структуры некоторого предприятия. Предусмотреть следующие классы «Человек», «Сотрудник», «Должность», «Подразделение». Предусмотреть иерархию классов.

**Примерные темы для итоговых проектов**

1. Постройте простую операционную систему. (Это технически сложный проект, который поможет углубить понимание работы компьютера и операционных систем).
2. Осуществление шифрования. Создайте инструмент, принимающий текстовую строку в качестве ввода и зашифровывающий ее, например, с помощью шифра Цезаря.
3. Калькулятор прибыли и трекер. Создайте калькулятор для отслеживания ежемесячного роста и снижения своих доходов. Можно руководствоваться чем-то вроде net worth worksheet от Чральза Шваба. Расширение: высылка отчета за предыдущие 12 месяцев 01 января каждого года.
4. Отслеживание расходов. Создайте простой интерфейс, который можно использовать для добавления и разбивки своих расходов по категориям. Генерируйте ежемесячный отчет, основанный на входящих данных, и напишите пользовательские уведомления вроде «тратишь слишком много на кофе… как всегда».
5. Игра «Жизнь» (Conway’s Game of Life). Игра «Жизнь» симулирует жизнь простых клеток, которые подчиняются алгоритмическим законам.
6. Напоминалка «Возьми зонтик». Постройте простое приложение, которое по утрам будет присылать на телефон уведомление о том, что нужно взять с собой зонтик (если в вашей местности ожидается дождь).

7. Постройте инструмент, автоматически создающий и обновляющий набор данных. Рекомендуемые наборы данных: статистика по вашей любимой спортивной команде, полеты в направлении, куда и вам бы хотелось отправиться, метеорологические данные региона, где вы живете. Набор данных должен нуждаться в регулярном обновлении по мере генерации новых данных, и это должно осуществляться автоматически. Например, когда результаты вашей спортивной команды публикуются на сайте, данные должны автоматически скрапиться и добавляться к набору данных.
8. Pixel art генератор. Постройте инструмент, принимающий изображение в качестве входящих данных и преобразует его в pixel art на выходе.
9. Движок для игры в крестики-нолики, который нельзя победить. Реализуйте стратегию, которая делает ничью худшим исходом игры.
10. Шахматный движок, способный играть в шахматы с человеком, используя GUI, совместимый с Universal Chess Interface, например, Xboard. В качестве примера посмотрите Stockfish. Чтобы упростить задачу можно сфокусироваться на поведении только какой-то части, например, коней.

## 6. Организационно – педагогические условия реализации программы

**Материально-техническое обеспечение.**

Занятия проходят в хорошо проветриваемом и освещённом классе, оборудованном мебелью, соответствующей санитарно-техническим требованиям и нормам возрастной физиологии (*парты, стулья, учительский стол и стул*).

Класс с рабочими местами учащихся и преподавателя, которые оборудованы компьютерами не менее 2 ГБ ОЗУ, процессор с тактовой частотой не менее 1.2 ГГц, диагональ мониторов не менее 12 дюймов, свободные 50 ГБ на накопителях, интернет не медленнее 1 Мбит/с.

**Программное обеспечение.**

- ОС — Windows/Linux/MacOS на усмотрение преподавателя.
- Любой современный браузер (например, Яндекс.Браузер, Google Chrome, Mozilla Firefox, Safari).
- Интегрированная среда разработки изучаемого языка программирования.

**Инструменты и расходные материалы.**

Канцелярские принадлежности, бумага, картриджи, и др.

## 7. Список литературы


1. Джошуа Блох. Java. Эффективное программирование. – М.: Лори, 2012.
2. Задачи по программированию. Под ред. С. М. Окулова, - М.: БИНОМ. Лаборатория знаний, 2016.
3. Зыков, С. В. Введение в теорию программирования. Курс лекций. Учебное пособие / С.В. Зыков. - М.: Интернет-университет информационных технологий, 2012.
4. Камаев В.А. Технология программирования. Учебник. - М.: Высшая школа, 2016.
5. Опалева, Э. А. Языки программирования и методы трансляции / Э.А. Опалева, В.П. Самойленко. - М.: БХВ-Петербург, 2015.
6. Роберт К. Мартин. Чистый код. Создание, анализ и рефакторинг. – СПб.: Питер, 2010.
7. Стив Макконнелл. Совершенный код. Мастер#класс / Пер. с англ. — М.: Русская редакция, 2010.
8. С. М. Окулов. Основы программирования. - М.: Бином. Лаборатория знаний, 2012.
9. Соколов А.П. Системы программирования: теория, методы, алгоритмы. - М.: ФиС, 2014.
10. Стенли Б. Липпман, Жози Лажойе,Барбара Э. Му. Язык программирования C++. Базовый курс, 5-е издание. — М.: Вильямс, 2017.
11. Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. Алгоритмы: построение и анализ, 3-е издание. — М.: Вильямс, 2013.


***Литература, рекомендованная учащимся***


1. Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. Алгоритмы: построение и анализ, 3-е издание. — М.: Вильямс, 2013.


***Ресурсы в Интернете***


1. Новый систематизированный толковый словарь по информационным технологиям [Электронный ресурс]: Государственная публичная научно-

техническая библиотека России. – Режим доступа: http://www.gpntb.ru/win/book, свободный.

2. Основы алгоритмизации и языков программирования [Электронный ресурс] – Режим доступа: http://bourabai.kz/alg/classification04.htm, свободный.
3. C++ с нуля [Электронный ресурс]: портал о программировании. – Режим доступа: https://code-live.ru/post/cpp-hello-world/#more, свободный.
4. Java — Учебник для начинающих программистов [Электронный ресурс] – Режим доступа: http://proglang.su/java, свободный.
5. Язык программирования Python 3 для начинающих и чайников [Электронный ресурс]: Python 3 для начинающих. – Режим доступа: https://pythonworld.ru/ свободный.
6. Быстрый старт со Scala для начинающих и не очень [Электронный ресурс] – Режим доступа: https://tproger.ru/articles/scala-tutorial-for-beginners/, свободный.
7. Язык программирования Swift. Русская версия [Электронный ресурс] – Режим доступа: https://habr.com/ru/post/225841/, свободный.