




Муниципальное автономное общеобразовательное учреждение

«Средняя общеобразовательная школа №2 п. Новоорск»

Новоорского района Оренбургской области

Центр «Точка роста»

<p>РАССМОТРЕНО на заседании МС</p> <p> /Рощина Е.А.</p> <p>Протокол №9 «17» августа 2023 г</p>	<p>СОГЛАСОВАНО Зам.директора по ВР</p> <p> /Шаранова Н.И.</p> <p>«17» августа 2023 г</p>	<p>УТВЕРЖДЕНО</p> <p>Директор МАОУ «СОШ №2 п. Новоорск» Горбунова И.М.</p> <p>Приказ № «17» августа 2023 г</p> 
---	---	--

**Дополнительная общеобразовательная (общеразвивающая)
программа творческого объединения
технической и цифровой направленности
«Программирование»**

Целевая аудитория: учащиеся 13--16 лет

Срок реализации: 1 год.

Разработала:
Рамзаева Ирина Валерьевна
учитель информатики

п. Новоорск
2023 г.

Пояснительная записка.

Программа дополнительного образования школьников «Программирование» имеет выраженную практическую направленность и способствует приобщению учащихся к алгоритмической культуре, а также дает им возможность изучить раздел на более углубленном уровне. Кроме того, выполняемые на занятиях задания способствуют развитию, и формированию у них аналитического мышления, в том числе умения анализировать, систематизировать, визуализировать информацию, работать с большими массивами данных, что является одним из приоритетных требований многих современных работодателей.

Целевая аудитория: дети старшего школьного возраста 13-16 лет, 7-9 класс.

Срок реализации программы: настоящая программа рассчитана на реализацию в течение 1 года, 1 час в неделю, 34 часа в год. Содержание данной программы, является начальной ступенью овладения комплексом минимума знаний и практических навыков, составляющих основу для последующей самостоятельной работы.

В качестве основного инструмента обучения выбран язык программирования Python. Его использование способствует формированию у учащихся более прочных и глубоких знаний, умений и навыков при составлении различных алгоритмов и написании программ со сложной структурой.

Программа дополнительного образования школьников «Программирование» имеет выраженную практическую направленность и способствует приобщению школьников к алгоритмической культуре, а также способствуют развитию творчества учащихся, и формированию у них аналитического мышления, умения анализировать, систематизировать, визуализировать информацию, работать с большими массивами данных, что является одним из приоритетных умений.

Основные цели курса:

- помочь детям узнать основные возможности программирования и научиться ими пользоваться в повседневной жизни;
- реализовать в наиболее полной мере интерес учащихся к изучению информатики и современных информационных технологий;
- способствовать формированию у учащихся навыков информационно-учебной деятельности на базе средств ИКТ для решения познавательных задач и саморазвития;
- раскрыть основные возможности, приемы и методы обработки информации разной структуры;
- способствовать развитию у учащихся информационной культуры.

Задачи курса:

Обучающие:

- способствовать освоению всевозможных методов решения задач, реализуемых с помощью ПО и языка программирования Python.
- научить применять структурный подход для решения практических задач с использованием компьютера,

- расширить знания, умения и навыки решения задач по программированию и алгоритмизации;
- сформировать у учащихся навыки практической исследовательской деятельности.

Развивающие:

- развивать стремление к самообразованию, обеспечить в дальнейшем социальную адаптацию в информационном обществе и успешную профессиональную и личную самореализацию;
- раскрыть креативные способности;
- способствовать развитию алгоритмического, творческого, логического и критического мышления.

Воспитательные:

- формировать информационную культуру учащихся;
- способствовать формированию активной жизненной позиции;
- воспитывать толерантное отношение в группе;
- добиться максимальной самостоятельности детского творчества;
- воспитывать собранность, аккуратность при подготовке к занятию;
- воспитывать умение планировать свою работу;
- сформировать интерес к профессиям, связанным с программированием.

Планируемые результаты

Личностные результаты

- 1) сформированность мировоззрения, соответствующего современному уровню развития науки и техники;
- 2) готовность и способность к образованию, в том числе самообразованию, на протяжении всей жизни; сознательное отношение к непрерывному образованию как условию успешной профессиональной и общественной деятельности;
- 3) навыки сотрудничества со сверстниками, детьми младшего возраста, взрослыми в образовательной, учебно-исследовательской, проектной и других видах деятельности;
- 4) эстетическое отношение к миру, включая эстетику научного и технического творчества;
- 5) осознанный выбор будущей профессии и возможностей реализации собственных жизненных планов; отношение к профессиональной деятельности как возможности участия в решении личных, общественных, государственных, общенациональных проблем.

Метапредметные результаты

- 1) умение самостоятельно определять цели деятельности и составлять планы деятельности; самостоятельно осуществлять, контролировать и корректировать деятельность; использовать все возможные ресурсы для достижения поставленных целей и реализации планов деятельности; выбирать успешные стратегии в различных ситуациях;

- 2) умение продуктивно общаться и взаимодействовать в процессе совместной деятельности, учитывать позиции других участников деятельности, эффективно разрешать конфликты;
- 3) владение навыками познавательной, учебно-исследовательской и проектной деятельности, навыками разрешения проблем; способность и готовность к самостоятельному поиску методов решения практических задач, применению различных методов познания;
- 4) готовность и способность к самостоятельной информационно-познавательной деятельности, включая умение ориентироваться в различных источниках информации, критически оценивать и интерпретировать информацию, получаемую из различных источников;
- 5) умение использовать средства информационных и коммуникационных технологий в решении когнитивных, коммуникативных и организационных задач с соблюдением требований эргономики, техники безопасности, гигиены, ресурсосбережения, правовых и этических норм, норм информационной безопасности.

Предметные результаты

- 1) владение системой базовых знаний, отражающих вклад *информатики* в формирование современной научной картины мира;
- 2) владение навыками *алгоритмического мышления* и понимание необходимости формального описания алгоритмов;
- 3) овладение понятием *сложности алгоритма*, знание основных алгоритмов обработки числовой и текстовой информации, алгоритмов поиска и сортировки;
- 4) владение стандартными приёмами *написания на алгоритмическом языке программы* для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ; использование готовых прикладных компьютерных программ по выбранной специализации;
- 5) владение *универсальным языком программирования высокого уровня* (по выбору), представлениями о базовых типах данных и структурах данных; умением использовать основные управляющие конструкции;
- 6) владение умением *понимать программы*, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования; умением анализировать алгоритмы с использованием таблиц;
- 7) владение навыками и опытом *разработки программ* в выбранной среде программирования, включая тестирование и отладку программ; владение элементарными навыками формализации прикладной задачи и документирования программ.

Содержание курса занятий кружка «Программирование»

Структура содержания курса данного кружка может быть определена следующими укрупнёнными тематическими блоками (разделами):

- введение в Python;
- алгоритмические структуры;
- подпрограммы;
- массивы;
- работа с графикой.

Раздел 1. Введение в Python. (6 часов)

Введение в Python. Структура программы на языке Python. Рекомендации по стилю записи программы, использование комментариев. Алфавит языка. Типы данных. Константы. Переменные. Организация ввода-вывода. Оператор присваивания. Общий вид программы на языке Python. Стандартные функции. Простейшая программа. Арифметические выражения. Правила записи арифметических выражений.

Раздел 2. Алгоритмические структуры (10 часов)

Организация программ разветвляющейся структуры. Условный оператор. Ветвление алгоритма на три рукава и более. Виды операторов цикла. Вложенные циклы.

Раздел 3. Подпрограммы (4 часа)

Подпрограммы (процедуры и функции), их описание и вызов в программе. Решение задач с математическим содержанием на использование подпрограмм. Файловые переменные. Ввод и вывод данных с использованием текстовых файлов.

Раздел 4. Массивы (10 часов)

Массивы (одномерные (линейные) и двумерные), различные способы их описания в программе. Обработка массивов (ввод и вывод элементов массива; поиск элементов в массиве; проведение математических операций с элементами массива; замена, удаление и вставка элементов в массиве; сортировка). Одномерные массивы: описание и ввод элементов, действия над ними. Поиск, замена в одномерном массиве. Сортировка массива. Способы сортировки. Понятие двумерного массива: описание и ввод элементов. Обработка элементов двумерных массивов. Сортировка массива. Способы сортировки.

Раздел 5. Итоговый проект (4 часа)

Выполнение обучающимися итогового проекта на языке Python.

Учебно-тематический план

№	название темы	количество часов	
		теория	практика
1	Введение в Python	2	4
2	Алгоритмические структуры	2	8
3	Подпрограммы	1	3
4	Массивы	2	8
5	Итоговый проект	1	3
	Итого:	8	26
		34	

Прогнозируемый результат:

По окончании изучения данного курса прогнозируется, что учащиеся будут обладать следующими знаниями, умениями и навыками:

- знать место языка Python среди языков программирования высокого уровня;
- знать особенности структуры программы, представленной на языке Python;
- знать основные операторы языка Python, их синтаксис;
- знать что такое алгоритм, свойства и типы алгоритмов, способы записи алгоритмов;
- знать назначение вспомогательных алгоритмов, технологии построения простых и сложных алгоритмов: метод последовательной детализации и сборочный (библиотечный) метод;
- уметь составлять линейные, ветвящиеся и циклические алгоритмы управления в среде учебных исполнителей;
- уметь выделять подзадачи; определять и использовать вспомогательные алгоритмы;
- знать правила описания процедур и построение вызова процедуры;
- решать различные задачи по программированию;
- иметь представление о таких структурах данных, как множество, запись, файл, стек, очередь, строка;
- знать, как формально определять в программе тип «массив»;
- знать свойства данных типа «массив»;
- создавать алгоритмы сортировки линейных числовых массивов и поиска в упорядоченном массиве;
- создавать программы и изображения в среде программирования Python.

КАЛЕНДАРНО-ТЕМАТИЧЕСКОЕ ПЛАНИРОВАНИЕ

№	Тема занятия	Кол-во часов
Раздел 1. Введение в Python (6 часов)		
1	ТБ при работе на компьютере. Введение в Python. Структура программы на языке Python. Рекомендации по стилю записи программы, использование комментариев. Алфавит языка.	1
2	Типы данных: целый и вещественный, логический и символьный. Константы. Переменные.	1
3	Организация ввода-вывода. Оператор присваивания.	1
4	Стандартные функции. Простейшая программа.	1
5	Арифметические выражения. Правила записи арифметических выражений.	1
6	Решение нестандартных задач	1
Раздел 2. Алгоритмические структуры (10 часов)		
7	Организация ветвлений в программах. Полное и неполное ветвление. Условный оператор.	1
8	Ветвление алгоритма.	1
9	Решение нестандартных задач.	1

10	Виды операторов цикла.	1
11	Цикл с постусловием.	1
12	Решение задач с использованием циклов.	1
13	Цикл с параметром.	1
14	Решение задач с использованием циклов с параметрами и ветвлением.	1
15	Вложенные циклы.	1
16	Решение задач с использованием вложенных циклов.	1
Раздел 3. Подпрограммы (4 часа)		
17	Подпрограммы (процедуры и функции), их описание и вызов в программе.	1
18	Решение задач с математическим содержанием на использование подпрограмм.	1
19	Файловые переменные. Ввод и вывод данных с использованием текстовых файлов.	1
20	Решение задач с использованием файловых переменных.	1
Раздел 4. Массивы (10 часов)		
21	Массивы (одномерные (линейные) и двумерные), различные способы их описания в программе.	1
22	Одномерные массивы: описание и ввод элементов, действия над ними.	1
23	Ввод и вывод элементов массива	1
24	Поиск элементов в массиве.	1
25	Проведение математических операций с элементами массива.	1
26	Замена, удаление и вставка элементов в массиве.	1
27	Сортировка элементов массива. Способы сортировки.	1
28	Понятие двумерного массива: описание и ввод элементов. Обработка элементов двумерных массивов.	1
29	Сортировка массива. Способы сортировки.	1
30	Решение нестандартных задач с использованием массивов.	1
Раздел 5. Итоговый проект (4 часа)		
31-34	Выполнение итогового проекта	4
Итого:		34

Формы организации образовательного процесса:

Групповые формы.

Обучающиеся работают в группах или в парах. Эту форму работы удобно использовать, при освоении новых программных средств, при работе над проектами, при недостаточном количестве компьютеров. Воспитанники обмениваются друг с другом информацией, вместе обсуждают задачу, оценивают решение каждого. Сверяют свои ответы и если допущены ошибки, то пытаются вместе найти ответ. Усвоение знаний и умений

происходит результативнее при общении учащихся с более подготовленными товарищами.

Дифференцированно - групповая форма.

Обучающиеся отличаются друг от друга умственной гибкостью, активностью, самостоятельностью мышления. Одни способны перебирать многообразие способов решения задач, чтобы найти верный путь решения. Другие привыкают работать по шаблону и не пытаются искать других подходов.

Для организации учебного процесса необходимо распределить обучающихся на несколько групп: по уровню знаний, интересам, способностям и подобрать задания в соответствии с выявленными уровнями знаний, интересами, способностями учащихся. Заданиями могут быть следующими: с различными условиями, допускающие одинаковые, с точки зрения информатики, решения; взаимодополняющие задания с различными условиями; уровневые взаимодополняющие задания.

Индивидуальные и парные формы.

При подборе заданий для индивидуальной самостоятельной работы учитываются уровни усвоения знаний учащимися: репродуктивный, репродуктивно - творческий, творческий. Работая один на один с компьютером (а точнее с программой), обучающийся в своем темпе овладевает знаниями, сам выбирает индивидуальный маршрут изучения учебного материала в рамках заданной темы занятия.

В парном обучении взаимодействие происходит между двумя учениками, которые могут обсуждать задачу, осуществлять взаимообучение или взаимоконтроль. Очень часто для учащегося помощь товарища оказывается полезнее, чем помощь учителя.

Режим работы занятий дополнительного объединения

«Программирование»

День проведения занятий	Время проведения занятий
понедельник	15 ⁰⁰ -15 ⁴⁵

Ожидаемые результаты и способы определения результативности

Учебный уровень достижений:

Обучающиеся должны знать:

- о концепциях и идеях структурного программирования;
- алгоритмические конструкции языка программирования;
- возможности инструментальных средств системы;
- основные приемы написания программ-приложений;
- требования к написанию и оформлению программ-приложений;
- типы данных и их представление в памяти компьютера, операции над данными основных типов;
- способы представления одномерных и двумерных массивов и строк;
- различие между текстовыми и бинарными файлами, особенности организации текстовых файлов;
- назначение и способы организации проектов.

Обучающиеся должны уметь:

- использовать все доступные источники (интерактивные компьютерные справочные системы, книги, справочники, технические описания) для самостоятельного решения задач с помощью компьютеров;
- составлять алгоритмы в словесной форме для решения разнообразных задач;
- применять метод пошаговой детализации при составлении алгоритмов;
- переводить алгоритмы на язык программирования;
- составлять алгоритмы и программы для новых методов решения задач;
- работать с различными структурами данных (массив, запись, файл, множество);
- решать поставленную задачу, реализовывать алгоритмические конструкции на языке программирования;
- правильно интерпретировать получаемые результаты в ходе тестирования и отладки программных продуктов.

Принципы построения программы

1) Принцип доступности – при изложении материала учитываются возрастные особенности детей, один и тот же материал по-разному преподаётся, в зависимости от возраста и субъективного опыта детей. Материал располагается от простого к сложному. При необходимости допускается повторение части материала через некоторое время.

2) Принцип наглядности – человек получает через органы зрения почти в 5 раз больше информации, поэтому на занятиях используются как наглядные материалы, так и обучающие, тестирующие программы.

3) Принцип развития выражается в возможности постоянного расширения и обновления системы задач, решаемых с помощью программирования и средств их достижения.

4) Принцип сознательности и активности – для активизации деятельности детей используются такие формы обучения, как занятия-игры, конкурсы, совместные обсуждения поставленных вопросов и свободное творчество.

5) Принцип индивидуализации базируется на том, что эффективность обучения прямо пропорциональна индивидуализации деятельности учащихся в ходе обучения.

6) Принцип практической направленности – в ходе обучения обучающиеся выполняют творческие проекты, разрабатывают собственные программы и внедряют их.

7) Принцип вариативности предоставляет педагогу возможность варьировать программу с учетом особенностей восприятия ее воспитанниками.

Формы подведения итогов реализации программы

Главный показатель – личностный рост каждого ребенка, его творческих способностей, превращение группы в единый коллектив, способный к сотрудничеству и совместному творчеству.

Проверка эффективности данного курса осуществляется через итоговое занятие. По окончании обучения по данной образовательной программе, учащиеся должны уметь создавать программы разного уровня сложности. Для оценки достижения обязательной подготовки целесообразно использовать дихотомическую шкалу, анализ работ обучающихся, определяющий творческий рост школьника, а также педагогическое наблюдение.

Условия реализации программы.

Важнейшим условием реализации программы является создание развивающей, здоровьесберегающей образовательной среды как комплекса комфортных, психолого-педагогических и социальных условий, необходимых для развития творческих интересов и способностей детей.

Материально-техническое обеспечение:

- компьютерный класс с 10 персональными компьютерами для обучающихся;
- локальная сеть с доступом в Интернет;
- проектор и демонстрационный экран;
- доска маркерная.

Программное обеспечение для компьютеров: язык программирования Python.

Формы аттестации и контроля

- защита проекта;
- зачетное занятие;
- выступление на конференции;
- участие в конкурсах различного уровня;
- участие в олимпиадах различного уровня.

ЛИТЕРАТУРА ДЛЯ ОБУЧАЮЩИХСЯ

1. Гейн А. Г, Сенокосов А. И. Информатика и информационные технологии. Учебник для 9 класса общеобразовательных учреждений. – М.: Просвещение, 2020 г. – 298 с.
2. Информатика. Задачник-практикум: В 2 т./ Под ред. И.Г. Семакина: Т.1. М.: БИНОМ. Лаборатория знаний, 2021 г.
3. Житкова О. А, Кудрявцева Е. К. Справочные материалы по программированию на языке Паскаль. – М.: «Интеллект-центр», 2020 г. – 80 с.
4. Ларина Э. С. Олимпиадные задания с решениями 9-11 классы. – В.: Учитель, 2020 г. – 111 с.
5. Ларина Э. С. Ларина «Создание программ на языке Паскаль» - В.: Учитель, 2018 г
6. Ляхович В. Ф. Основы информатики. – Р.: ЕНИКС, 2020 г.
7. Окулов С.М. Основы программирования. – М.: Юнимедиастилл, 2019 г.
8. Ушаков Д. М., Юркова Т. А. «Паскаль для школьников» - М.: Питер, 2021 г.
9. Чернов А. А., Чернов А. Ф. «Контрольные и самостоятельные работы по программированию» - В.: «Учитель», 2020 г.

Диагностика результатов

Система отслеживания результатов программы

Для определения исходного уровня развития алгоритмического мышления учащихся мною была разработана входная диагностика.

Диагностика № 1

1. Решить задачу и объяснить ход рассуждений:

1.1. Имеется пять монет, среди которых одна фальшивая (легче других). Придумайте способ нахождения фальшивой монеты за минимальное число взвешиваний на чашечных весах без гирь.

1.2. Из четырех внешне одинаковых деталей одна отличается по массе от трех остальных, однако неизвестно, больше ее масса или меньше. Как выявить эту деталь двумя взвешиваниями на чашечных весах без гирь?

2. Как с помощью 7-литрового ведра и 3-литровой банки налить в кастрюлю ровно 5л воды?

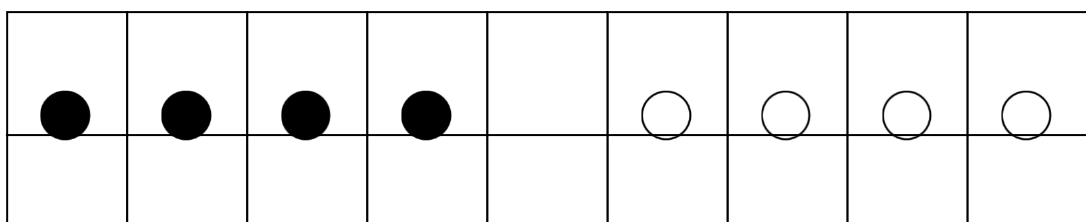
3. Имеются два кувшина емкостью 3л и 8л. Запишите алгоритм, выполняя который можно набрать из реки 7 л воды (другими емкостями не пользоваться).

4. Решить задачу и объяснить ход рассуждений:

По каналу плывут одна за другой лодки *A* и *B*. Навстречу им движутся одна за другой лодки *B* и *Г*. Канал настолько узок, что две лодки в нем разъехаться не могут, однако в нем с одной стороны есть небольшой залив, куда может поместиться только одна лодка. Могут ли лодки разъехаться так, чтобы продолжать свой путь в прежнем порядке?



1. Четыре белые и четыре черные шашки расположены так, как показано на рисунке:



Требуется переставить белые шашки на клетки с номерами 1, 2, 3, 4, а черные – на клетки с номерами 6, 7, 8, 9 с соблюдением следующих условий:

- 1) каждая шашка может перескочить на ближайшую клетку или через одну клетку, но не дальше;
- 2) никакая шашка не должна возвращаться на клетку, где она уже побывала;
- 3) в каждой клетке не должно быть более одной шашки;
- 4) начинать ходы надо с белой шашки.

Критерии	Уровни		
	Высокий	Средний	Низкий
1. Степень осознанности приемов алгоритмического мышления	-глубокое осознанное владение алгоритмическим языком, алгоритмическими структурами; - умение видеть в примерах алгоритмическую структуру; -осознанное использование алгоритмических знаний и умений при решении задач;	- фрагментарное осознанное владение алгоритмическим языком, алгоритмическими структурами; - фрагментарное осознанное использование алгоритмических знаний и умений при решении задач;	-слабая осознанность знаний алгоритмического языка; -слабое осознанное использование алгоритмических знаний при решении задач.
2. Степень владения приемами алгоритмического мышления.	-свободное использование знаний алгоритмического языка при решении проблемных задач из различных дисциплин.	-частичные затруднения при решении проблемных задач из различных дисциплин.	-формальное владение приемами алгоритмического мышления при решении проблемных задач.
3. Тезаурус алгоритмического мышления	-системность знаний, способность видеть альтернативные решения; -самостоятельность в дальнейшем развитии алгоритмического мышления.	-системность знаний; -частичные затруднения при выборе альтернативных решений; - для получения новых знаний требуется иногда помощь учителя.	-несистемный набор знаний; - затруднения при выборе альтернативных решений; - получение новых знаний с помощью учителя.

Для каждого воспитанника выставлялась оценка (от 1 до 6 баллов) по каждому критерию (приложение 1), затем вычислялся обобщенный показатель развития алгоритмического мышления и по оценкам определялся соответствующий уровень: низкий уровень (1-3 баллов), средний (4-5 баллов), высокий (6 баллов)

Методика обучения алгоритмизации.

Задачи на составление алгоритмов можно разбить на четыре типа:

- найти ошибку в алгоритме;
- определить, каков результат выполнения алгоритма;
- сформулировать условие задачи;
- построить математическую модель, составить алгоритм.

Рассмотрим примеры задач первого типа:

1) Найти ошибку в алгоритме:

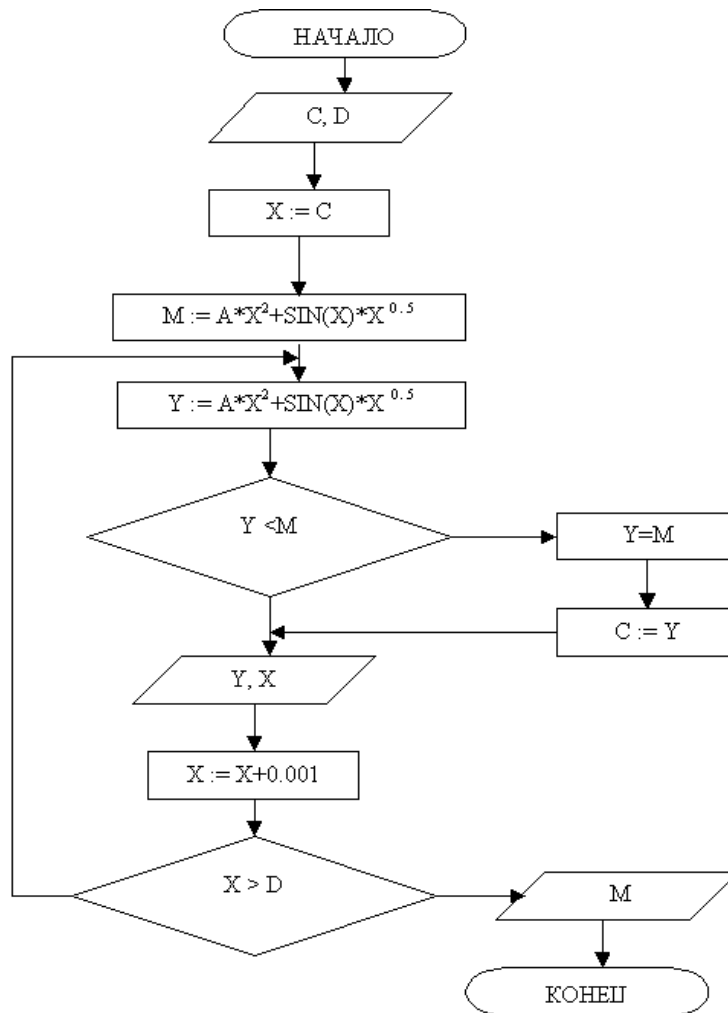
- а) значение b умножить на два;
- б) a присвоить значение b ;
- в) вывести значение b на экран.

2) Злоумышленник поменял местами действия в алгоритме вычисления среднего арифметического квадратов трех чисел.

- а) Присвоить a значение $(a^2+b^2+c^2)/3$.
- б) Запросить a, b, c .
- в) Сообщить "Среднее арифметическое квадратов равно".
- г) Сообщить a .

Восстановить правильный порядок чисел.

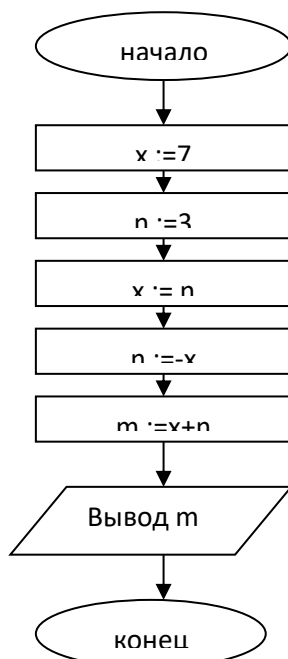
3) Определите местонахождение ошибок в алгоритмическом решении следующей задачи: найти минимальное значение функции $Y=A*X^2+\sin(X)*X^{0.5}$, для X изменяющемся на отрезке $[C,D]$ с шагом $0,01$.



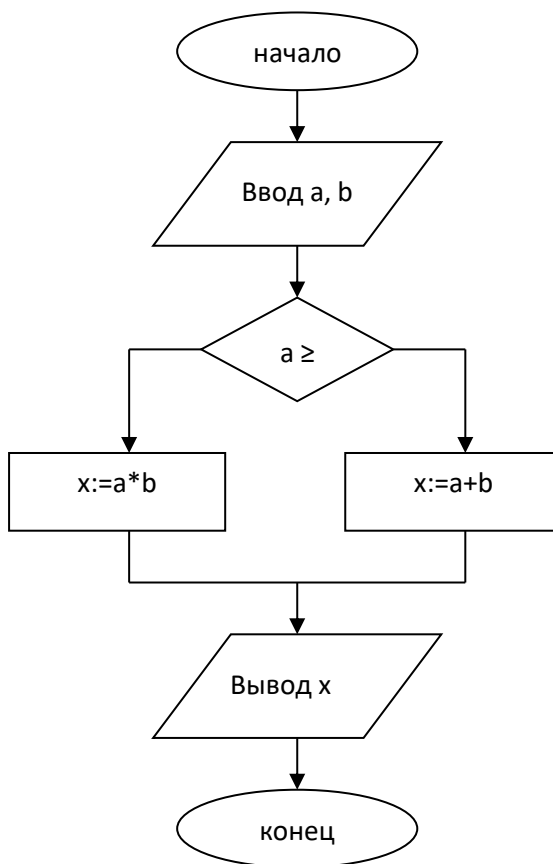
Для развития алгоритмического мышления воспитанников было важно постоянно тренироваться в упражнениях, которые имели алгоритмические ошибки. Важным этапом использования таких упражнений был анализ ошибок, который ставил обучающихся в рефлексивную позицию и способствовал процессу развития алгоритмического мышления воспитанников.

Рассмотрим несколько заданий, относящихся ко второму типу.

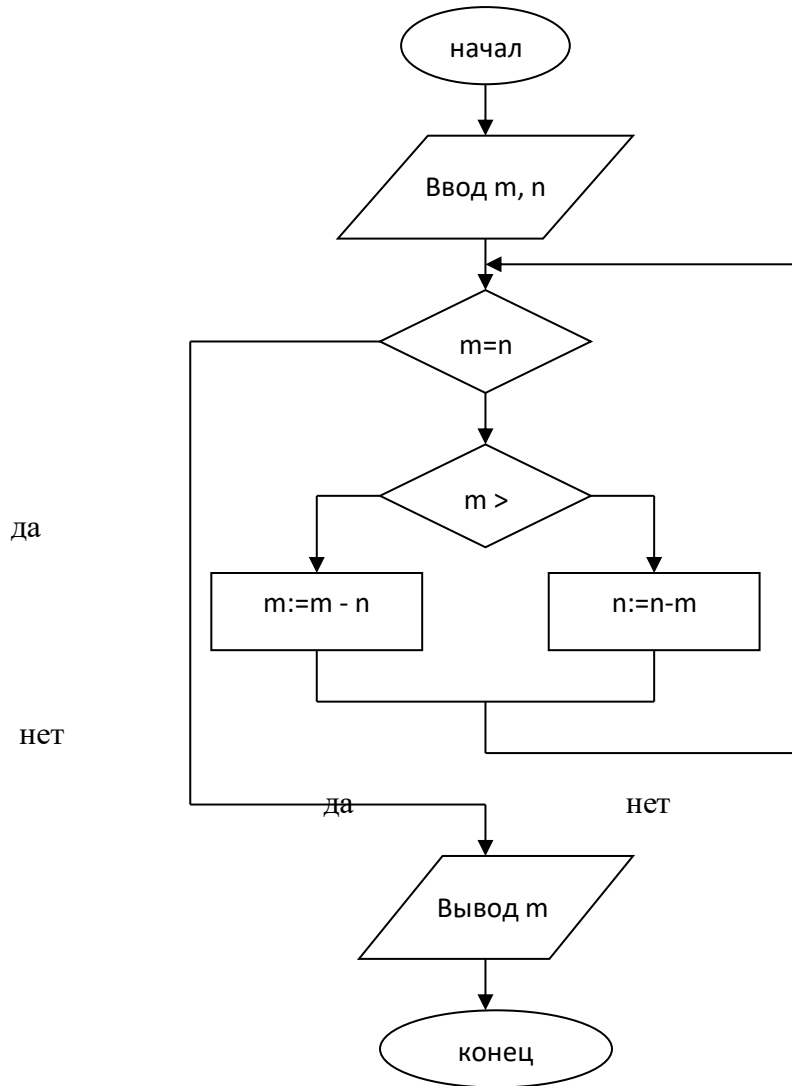
1) Определите, что получится в результате выполнения алгоритма, описанного блок-схемой.



2) При исходных данных $a = 5$, $b = 4$ определите результат выполнения алгоритма, изображенного в виде блок – схемы.



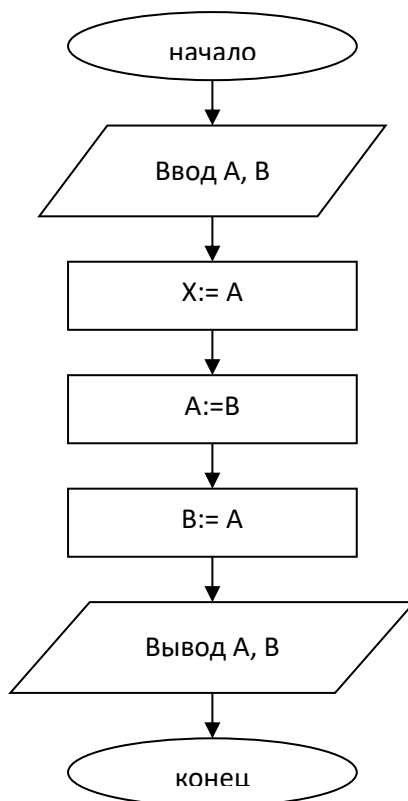
3) При исходных данных $m = 34$, $n = 85$ определите результат выполнения алгоритма, изображенного в виде блок – схемы.



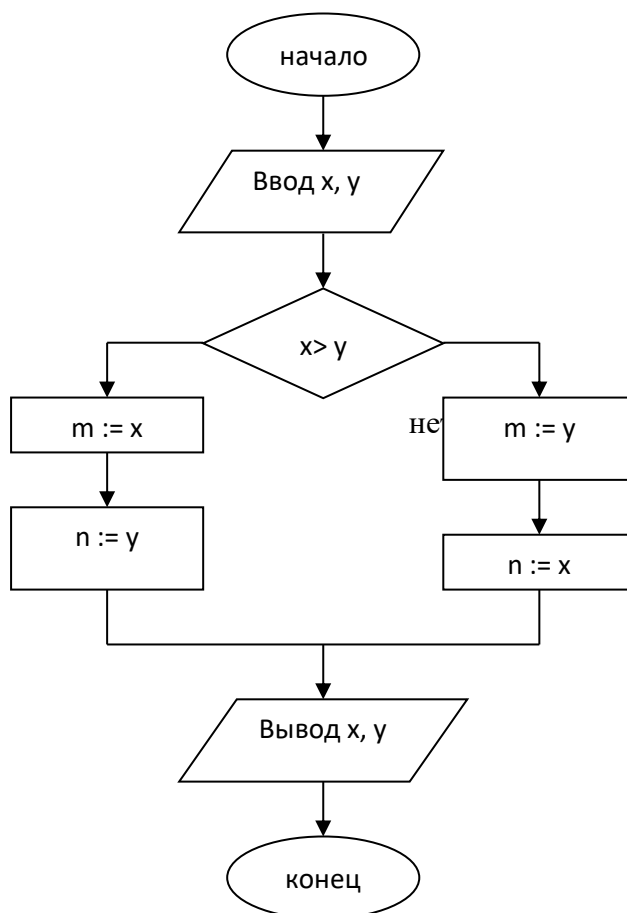
Задания, относящиеся к третьему типу:

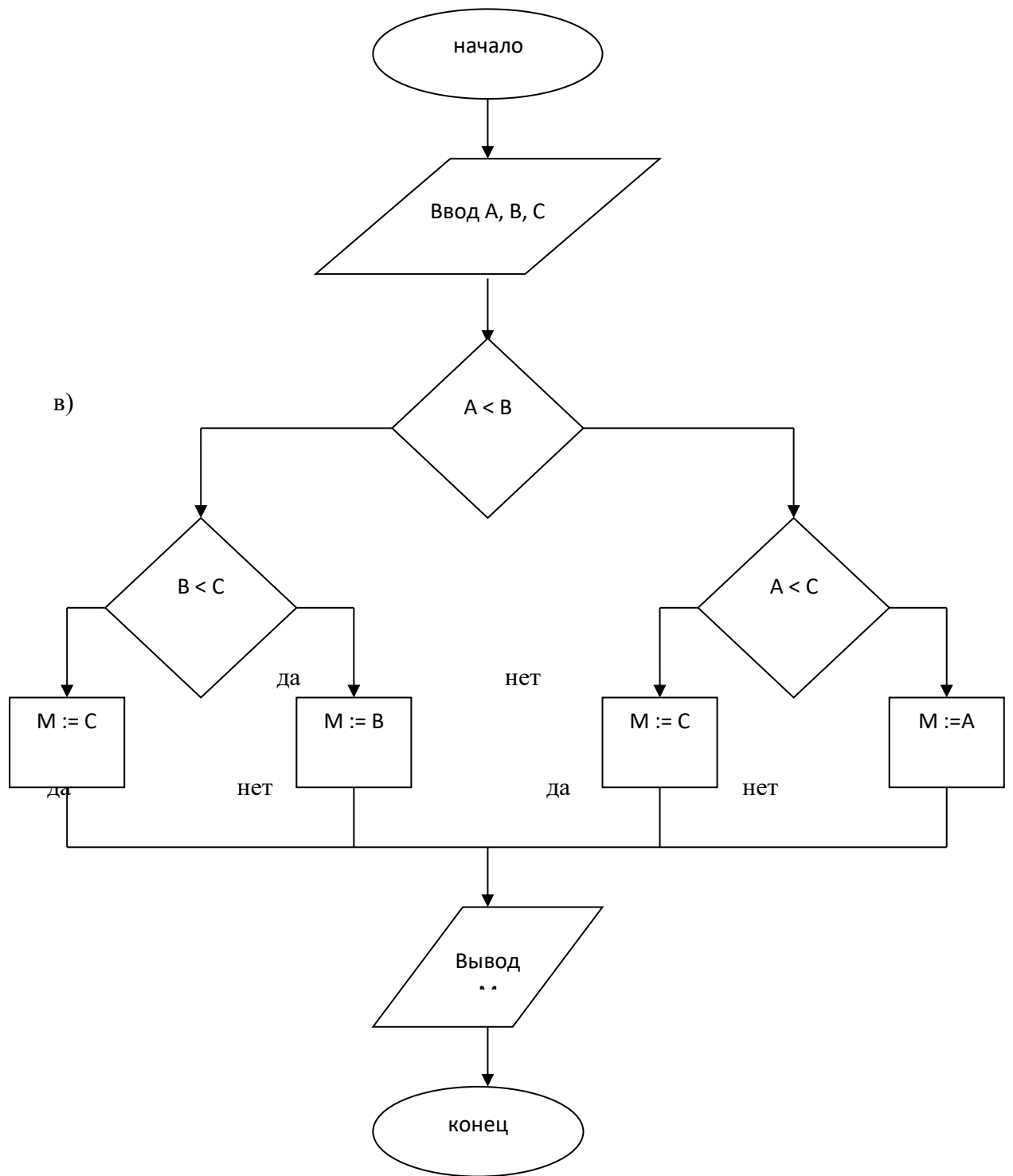
По предложенной блок-схеме сформулируйте условие задачи.

1. а)



б)





В качестве примера задачи четвертого типа можно использовать алгоритм игры Баше.

Правила игры определены так: в игре используются 11, 16, 21, 26 предметов. За один ход можно брать от 1 до 4 предметов. Проигрывает тот игрок, который берет последний предмет.

После того как ученики поиграли в игру по этим правилам, можно предложить им несколько заданий аналитического характера на тему игры Баше. Задания могут быть предложены в качестве домашней работы.

- 1) «Разгадать загадку» алгоритма, т.е. объяснить, почему второй игрок всегда выигрывает?
- 2) Составить алгоритм, по которому игрок, делающий первый ход, может выиграть только в том случае, если соперник не знает выигрышной тактики.
- 3) Попробуйте провести математический анализ игры Баше в общем случае для N камней. Определите правила игры (т.е. сколько камней можно брать за один ход), при котором имеется выигрышный алгоритм. Опишите этот алгоритм в виде последовательности команд.

Использование подобных игр приводит к выработке у воспитанников умения анализировать, сравнивать, т.е. логически и алгоритмически мыслить, обосновывать выдвигаемые положения.

Методика обучения программированию.

Для решения задачи с помощью компьютера (в средах программирования) знакомя учащихся с этапами решения задач (к этапам, которые присутствуют при составлении алгоритмов, добавляются этапы, необходимость которых обусловлена тем, что реализация модели будет осуществляться техническим средством).

Этапы решения задачи на компьютере с использованием среды программирования.

№	Название этапа	Содержание этапа
1	Постановка задачи Формализация условий задачи	<ul style="list-style-type: none">• Что дано?• Что требуется?• Что будет считаться результатом (решением задачи)?• Каким условиям должны удовлетворять исходные данные?• Каким условиям должен удовлетворять результат?
2	Выбор метода решения	Выбор метода решения зависит от точности, с которой мы хотим получить результат.

3	Построение математической модели	Установление связи между исходными данными и результатом (формулы, соотношения, неравенства и т. п.). Модель всегда основана на некотором упрощении. При построении модели также учитывается точность, с которой мы хотим получить результат.
4	Составление сценария работы с ЭВМ	<p>Определение:</p> <ul style="list-style-type: none"> • порядка ввода данных, • формы представления промежуточных результатов, • формы вывода результата, • реакции программы на вводимые команды и данные.
5	Разработка (конструирование) алгоритма	<p>При разработке алгоритма следует учитывать:</p> <ul style="list-style-type: none"> • имеющиеся ресурсы памяти, • возможности компьютера, • какие операции доступны для исполнения.
6	Перевод алгоритма в программу	Программа — алгоритм, закодированный по правилам выбранного языка программирования. Один и тот же алгоритм на разных языках программирования будет записан по-разному.
7	Ввод программы в память компьютера	Работа во встроенном текстовом редакторе среды программирования.
8	Отладка программы	Отладка — устранение синтаксических и семантических (смысловых) ошибок. Такого рода ошибки обнаруживаются на этапе компилирования программы.
9	Тестирование программы	Тестирование — устранение логических ошибок. Неоднократное выполнение программы с такими исходными данными, при которых результаты работы легко проверить.
10	Получение и анализ результатов	Получение результата решения задачи, для которой создавался алгоритм.

Учащиеся должны усвоить, что многократное тестирование еще не гарантирует безошибочной программы, а единственная ошибка при тестировании говорит о наличии ошибки в программе (или алгоритме).

Ни одну более или менее сложную программу нельзя считать правильной и процесс ее написания законченным, если он не проверен путем исполнения. Велика

обучающая роль исполнения программ - это, в конце концов, приводит к сознательному и прочному усвоению конструкций и правил алгоритмического языка. Преподавателю надо знать, что привить воспитанникам навыки программирования можно только путем обучения учащихся самостоятельно исполнять их.

Подобно задачам по теме "Алгоритмизация", задачи в теме "Программирование" можно разбить на следующие типы:

- исполнение программы;
- найти ошибку в программе,
- определить, каков результат выполнения программы,
- усложнение задачи;
- построить математическую модель, составить алгоритм, написать программу, проверить ее.

Задачи с заданием "Найти ошибку в программе" или "Определить, каков результат выполнения программы" рекомендуется предлагать учащимся систематически в качестве общего задания для группы в начале урока (аналогично "устному счету" в младших классах). Выполнение задания фронтально позволит преподавателю:

- с первых минут активизировать внимание воспитанников на занятии,
- еще раз проговорить алгоритм выполнения программы для воспитанников, которые не поняли эту тему на первых занятиях,
- обратить внимание учащихся на типичные ошибки, выявленные преподавателем после проведения диагностической работы.

Задания на нахождение ошибок:

1) Найти ошибку в программе. Запиши её на языке программирования, который изучал

```
program 1;  
var a, b, c, S, p: real;  
begin  
write ('Введите длины сторон треугольника:');  
readln (a, b, c);  
p := (a + b + c)/2;  
S :=sqrt (p*(p - a)*(p - b)*(p - c) );  
End.
```

2) Найдите ошибки в записи условного оператора (типы данных считать допустимыми):

- a) if A=B and A=C then R:=2; else R:=5;
- б) if (A=B) or C Then R:=5 else R:=5;

в) if A=B then begin R:=2; S:=5 end else R:=5; S:=2;

г) if A>10 then R:=2 else if A>15 then R:=6;

3) Найти ошибку в программе, запиши на языке программирования, который изучал:

```
Var k, N, S: integer;  
  
Begin  
  
  Readln (N);  
  
  For k := 1 to N do S := S + k;  
  
  Writeln ('S = ', S);  
  
End.
```

Задания на определение результата выполнения программы можно, кроме того, предлагать воспитанникам для самостоятельного выполнения. Особенно они эффективны на первых занятиях знакомства с командами языка программирования.

Например,

1) x=1.2; y=x*2;

y=y-5*x;

x=abs(y);

print ('x=', x, 'y=', y);

2) Что будет выведено на экран дисплея после выполнения приведенных ниже операторов?

A=123.45; b=0.0005;

print (a, b);

print ('a=', a, 'b=', b);

Задания на усложнение программы необходимо предусматривать на любом занятии. Для воспитанников была подобрана специальная система постепенно усложняющихся задач. Подобная организация работы воспитанников позволяет стимулировать и поддерживать их умственную активность.

С помощью специального отбора дидактического материала, происходило усвоение теоретических знаний, включивших в себя знания алгоритмического языка, основных алгоритмических конструкций. Проблемные задания способствовали формированию познавательного интереса.

Для развития алгоритмического мышления воспитанников было важно постоянно тренироваться в упражнениях, которые имели ошибки. Задания с ошибками стимулировали развитие алгоритмического мышления, так как требовали внимания к излагаемому материалу, активизировали деятельность воспитанников.

Задания для проверки знаний учащихся по разделу программирование.

**Диагностическая работа №1.
Программирование линейных алгоритмов.**

Вариант 1.

1. Вычислить и напечатать длину окружности и ее радиус, если площадь соответствующего круга равен $31,4 \text{ см}^2$.
2. Вычислить и напечатать результат выражения
$$x = \frac{a+b}{ab}, \quad a=10, b=5$$
3. Вычислить длину стороны A , площадь S и периметр P квадрата, описанного около окружности радиуса R .
4. Коническая куча зерна имеет высоту H м и длину окружности основания C м. Составить программу определения массы зерна, если масса 1 м^3 его равна M кг.

Вариант 2.

1. Вычислить и напечатать длину стороны и длину диагонали квадрата, если его периметр равен $10,4 \text{ см}$.
2. Вычислить и напечатать результат выражения
$$x = \frac{5(a+2d)}{a+d} \quad \text{при } a=15, d=5$$
3. Вычислить длину стороны A , площадь S и периметр P квадрата, описанного около окружности радиуса R .
4. даны высота H и площади $Q1$ и $Q2$ верхнего и нижнего основания усеченной пирамиды. Написать программу определения объема усеченной пирамиды.

**Диагностическая работа №2
Программирование разветвляющихся алгоритмов.**

Вариант 1.

1. Написать программу, запрашивающую возраст пользователя. Если ему не менее 18 лет, сообщите, сто он имеет право голосовать, в противном случае вычислите, через сколько лет ему будет предоставлено это право.
2. Написать программу, вычисляющую значение функции
$$y = \frac{ab}{a^2 - b^2}$$
3. Написать программу, определяющую существует ли треугольник, длины сторон которого равны числам A , B и C .
4. В ЭВМ поступают данные о количестве полных лет трех призеров спартакиады. Написать программу, выбирающую и печатающую возраст самого старшего призера.

Вариант 2.

1. Написать программу, уменьшающую четное число X в два раза.
2. Написать программу, вычисляющую значение функции
$$z = \frac{x+y}{x^2 + y^2}$$
3. Написать программу, определяющую по длинам сторон A , B и C , является ли треугольник прямоугольным.
4. В ЭВМ поступают результаты соревнований по плаванию для трех спортсменов. Написать программу, выбирающую и печатающую лучший результат.

**Диагностическая работа №3.
Программирование циклических алгоритмов.**

Вариант 1.

1. Составить программу вычисления суммы первых N членов натурального ряда.
2. Не пользуясь операцией умножения, составить программу умножения натурального числа A на натуральное число B .
3. В первый час работы рабочий изготавливает 25 деталей, за каждый последующий час на 3 детали больше, чем за предыдущий. Подсчитать, сколько рабочий изготовит деталей за 8 часов работы.

Вариант 2.

1. Найти сумму и произведение всех целых чисел от 1 до 20.
2. Составить программу возведения числа A в натуральную степень N .
3. В кинотеатре 30 рядов кресел. В первом ряду 20 кресел, в каждом последующем на 2 кресла больше, чем в предыдущем. Стоимость одного места в первом ряду – 5 рублей, а в каждом последующем – на 1 рубль больше, чем в предыдущем. Подсчитать выручку кинотеатра за один сеанс при целиком заполненном зале.

Диагностическая работа №4.

Программирование вспомогательных алгоритмов.

Вариант 1.

1. Написать программу вычисления значений функции при заданных значениях аргумента. Вычисление y оформить во вспомогательном алгоритме.
$$y = x^2 + x + 1, \quad x \in \{5; 9; 15\}$$
2. Разработать собственный шрифт. Написать программу, выводющую на экран слово «КАССА». Для вывода букв слова использовать вспомогательные алгоритмы для букв К, А и С.
3. Написать программу вычисления значения выражения
$$\frac{a! + b!}{c!}$$

Вариант 2.

1. Написать программу вычисления значений функции при заданных значениях аргумента. Вычисление y оформить во вспомогательном алгоритме.
$$y = \frac{1}{2x}, \quad x \in \{-1; 1,8; 2\}$$
2. Разработать собственный шрифт. Написать программу, выводющую на экран слово «БАНАН». Для вывода букв слова использовать вспомогательные алгоритмы для букв Б, А и Н.
3. Написать программу вычисления значения выражения
$$\frac{n!}{m!(n-m)!}$$

Диагностическая работа №5

Одномерные массивы

Вариант 1

1. С помощью датчика случайных чисел на интервале от 150 до 300 получить целочисленный массив, состоящий из 135 элементов.
2. Дан целочисленный массив заданный датчиком случайных чисел в интервале (100). Требуется умножить отрицательные значения элементов этого массива на 20.

Вариант 2

1. Составьте программу вычисления произведения тех значений элементов массива, которые меньше **15**.
2. Дан одномерный массив. Из элементов массива, равных **100**, сформировать массив **b**., а из элементов меньших **33**, сформировать массив **c**.

Двумерные массивы


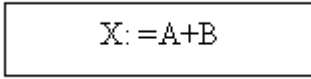
Вариант 1

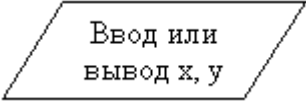
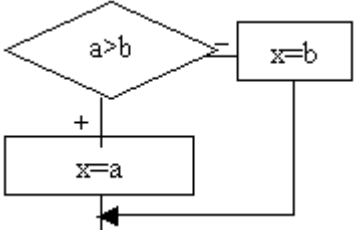
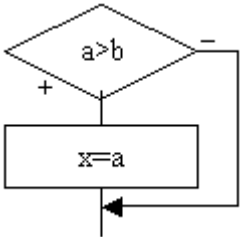
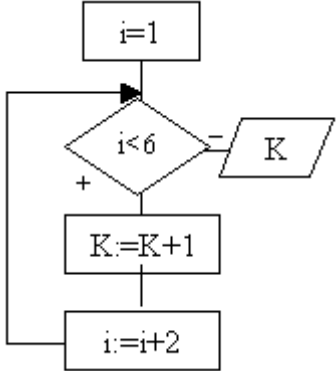
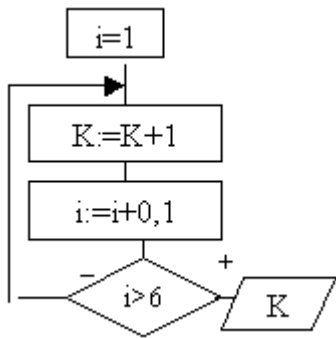
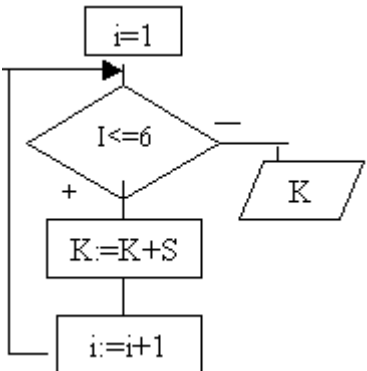
1. Из целочисленного массива $A(n,m)$ сформировать массив $B(n)$, в котором элемент $b[i]$ равен количеству положительных элементов i -й строки массива $A(n,m)$.
2. В целочисленном массиве $A(n,m)$ для каждого столбца найдите среднее арифметическое максимального и минимального элементов.
3. Удалите из массива $A(n,m)$ все столбцы, состоящие только из положительных элементов.
4. Вставьте в массив $A(n,m)$ после первой строки, первый элемент которой отрицателен, нулевую строку.

Вариант 2

1. Из целочисленного массива $A(n,m)$ сформировать массив $B(n)$, в котором элемент $b[i]$ равен 1, если среди элементов i -ой строки есть минимальный элемент массива $A(n,m)$ или 0, если минимального элемента в строке нет.
2. В целочисленном массиве $A(n,m)$ для каждой строки найдите среднее арифметическое положительных чётных элементов.
3. Удалите из массива $A(n,m)$ первую строку с максимальным элементом (считается, что в массиве несколько максимальных элементов).
4. Вставьте в массив $A(n,m)$ после каждого столбца из положительных элементов столбец из 0.

Дидактический материал, используемый на занятиях Таблица соответствия алгоритмических и программных фрагментов

Фрагменты блок-схем алгоритмов	Назначение	Соответствующие фрагменты программ на языке Python
 <p style="text-align: center;">Начало</p> <p style="text-align: center;">Конец</p>	<p>Начало и конец алгоритма.</p>	
 <p style="text-align: center;">$X := A + B$</p>	<p>Блок обработки (в нем вычисляются новые значения или</p>	

	производится вызов подпрограмм).	
	Ввод исходных данных или вывод результатов.	
	Ветвление полное. Если значение переменной a больше b , то выполняется $x=a$, иначе $x=b$.	
	Ветвление неполное. Если значение переменной a больше b , то выполняется $x=a$.	
	Цикл с предусловием. Пока значение условия $i < 6$ истинно выполняется тело цикла, то есть действия $K=K+1$ и $i=i+2$. Переменная i определяет количество повторений и называется счетчиком цикла .	
	Цикл с постусловием. Пока значение условия $i > 6$ ложно выполняется тело цикла, то есть действия $K=K+1$ и $i=i+0,1$. Переменная i определяет количество повторений в цикле и называется счетчиком цикла .	
	Цикл с постоянным приращением счетчика. В этом цикле изменение счетчика цикла i происходит только на единицу. Пока значение счетчика цикла меньше или равно 6 выполняется тело цикла, то есть действия $K=K+S$ и $I=i+1$.	

