Программа вступительного испытания по информатике для поступающих в 10 класс информационно-технологического профиля ГБОУ Школа № 1535

01. Цель вступительного испытания

Цель вступительного испытания – проверка следующих умений и навыков абитуриентов:

1. навык алгоритмического мышления;
2. умение составить алгоритм решения задачи и реализовать его на одном из языков программирования;
3. навык смыслового чтения текста, умение формализовать задачу, записанную в неформальном виде.

02. Регламент проведения вступительного испытания

1. Испытание состоит из 5 задач, решаемых на компьютере на выбранном абитуриентом языке программирования.
2. Испытание проводится на школьных компьютерах с использованием автоматизированной тестирующей системы (аналогичной используемой на олимпиадах по информатике и на сайте informatics.msk.ru). Использовать для написания программ свои компьютеры не разрешается.
3. На компьютерах будут доступны следующие языки программирования и среды разработки:
    1. Pascal (Pascal ABC.NET)
    2. Python 3 (IDLE, Wing IDE, PyCharm)
    3. C/C++ (Visual Studio, Code Blocks)
    4. Java (Eclipse)
4. Продолжительность испытания – 120 минут. Перед началом мониторинга у абитуриента будет возможность проверить наличие необходимых сред программирования и их корректную работу, а также познакомиться с тестирующей системой и с принципом её работы.
5. Во время мониторинга нельзя общаться с другими абитуриентами. Все вопросы можно задавать только организаторам в аудитории. Для того, чтобы задать вопрос, необходимо поднять руку и дождаться, когда к подойдет организатор.
6. Во время проведения испытания нельзя пользоваться ресурсами сети Интернет, а также любыми записями в бумажном или электронном виде.
7. За нарушение правил пунктов 5 и 6 учащийся удаляется с мониторинга, а его результат аннулируется.

03. Программа вступительного испытания. Проверяемые знания

1. **Переменные**. Типы данных (целые числа, вещественные числа, строки). Ввод и вывод данных различных типов.
2. **Вычислительные алгоритмы**. Арифметические операции: сложение, вычитание, умножение, вещественное и целочисленное деление, получение остатка, квадратный корень.
3. **Целочисленные алгоритмы**:
    1. выделение отдельных цифр десятичного числа;
    2. представление значения времени, выраженного в секундах, в часы, минуты и секунды (и наоборот – время, выраженное в часах, минутах и секундах, переводится в секунды);
    3. преобразования различных единиц измерения: сантиметры в метры и километры, граммы в килограммы, биты в килобайты и байты и т. д.
4. **Условный оператор**. Краткая и полная форма записи условного оператора. Вложенные условия. Логические операции в записи условий: «И», «ИЛИ», «НЕ».
5. **Циклы**. Цикл с параметром (for), цикл с предусловием (while) , цикл с постусловием (при наличии в выбранном языке программирования).

6. **Обработка числовых последовательностей без запоминания чисел в массиве (за один проход):**
    1. подсчёт количества чисел, удовлетворяющих заданным условиям;
    2. определение наличия в последовательности чисел элемента, удовлетворяющего заданным условиям;
    3. поиск минимального и максимального значений в последовательности;
    4. подсчёт количества чисел, равных минимальному или максимальному значению;
    5. определение номера первого или последнего числа, равного минимальному или максимальному значению.
7. **Массивы.** Ввод и вывод массивов. Обработка массивов в цикле.
8. **Алгоритмы обработки массивов:**
    1. подсчёт количества элементов массива, удовлетворяющих заданному условию;
    2. проверка наличия в массиве заданного элемента;
    3. определение минимального/максимального элемента;
    4. подсчёт количества элементов, равных минимальному или максимальному элементу;
    5. определение номера первого или последнего элементов, равных минимальному/максимальному значению;
    6. поиск последовательности максимальной длины, состоящей из одинаковых элементов (например, поиск в массиве чисел максимального количества подряд идущих единиц).

В задачах не будет обработки строковых данных, все входные данные будут являться целыми или вещественными числами (хотя, конечно, никто не запрещает рассматривать число как строку). Это сделано по двум причинам. Во-первых, серьезные алгоритмы обработки строк выходят за рамки программы основного образования. Во-вторых, работа со строками достаточно сильно различается в разных языках программирования, а также часто требует применения справочных материалов (помнить наизусть все функции и их аргументы может быть затруднительно).