Положение к проведению учебно-диагностического входного контроля в целях набора 8-го математического IT-профиля

Оглавление

Общие положения

Отбор в учебные профили, связанные с углубленным изучением информатики и программирования, предусматривает успешное прохождение учебно-диагностических контрольных мероприятий, включая предмет «Информатика и ИКТ» (в дальнейшем – «Информатика»).

Для предмета информатики предусмотрено проведение экзамена или зачёта с оценкой (в дальнейшем принято общее наименование «зачёта») продолжительностью **150** минут.

Зачёт состоит из двух диагностических частей: теоретической и практической. На зачёте учащийся демонстрирует как фактическое знание содержательного теоретического материала, так и практические умения создания, применения и использования алгоритмов решения задач, понимание принципов построения и исполнения основных алгоритмических структур, а также знание отдельно выносимых основных алгоритмов, широко применяющихся в практике программирования.

В качестве базового языка программирования для сдачи экзамена выступает язык Python версии не ниже 3-ей, однако допускается использование иных распространённых языков программирования, например, языка C++, Pascal или Java. Ввиду различных особенностей, присущих разным языкам программирования, учащийся может попросить получить задания, записанные на одном из перечисленных выше языков программирования, аналогичные заданиям для языка Python.

Тематическое содержание

Тематическое содержание охватывает все основополагающие аспекты темы «**Алгоритмизация и программирование**», соответствующие кодификаторам ФГОС:

- **1.2 «Алгоритмизация и программирование;**
- **1.3 «Основы логики», п. 1.3.1 «Алгебра логики»**
- **2.8 «Технологии программирования»**

Основными блоками тематического содержания являются следующие:

1. Алгоритмы и их свойства
2. Алгоритмические структуры:
    a. Линейная структура

   b. Структура ветвления
   c. Циклическая структура

Полная версия тематического содержания, проверяемого на зачёте, представлена ниже:

1. Понятие алгоритма и его свойства
2. Способы записи алгоритмов:
   a. Словесная форма
   b. Блок-схемная форма
   c. Программный код
3. Алгоритмические структуры:
   a. Линейная структура
   b. Структура ветвления
   c. Циклическая структура
4. Линейные алгоритмы. Операторы ввода/вывода и диалоговые программы
5. Операторы целочисленного деления:
   a. Оператор «//»
   b. Оператор «**%**»
6. Условный оператор (оператор «IF»):
   a. Формы условного оператора
   b. Логические условия и логические операции
   c. Логический тип данных
   d. Составление сложных логических выражений для решения логических задач
7. Циклический алгоритм. Общие принципы построения и структурные составляющие любого цикла
8. Операторы циклов языка Python:
   a. Оператор «**FOR**»
   b. Оператор «**WHILE**»
9. Вложенные циклы

Основные алгоритмы

При подготовке к зачёту необходимо изучить, понимать и уметь применять на практике некоторые распространённые и широко используемые алгоритмы, полный список которых перечислен ниже:

1. Алгоритм нахождения факториала любого целого положительного числа

2. Алгоритм возведения любого целого положительного числа в любую целую положительную степень, не используя стандартные функции языка программирования
3. Алгоритм нахождения суммы, количества и среднего арифметического чисел последовательности, отвечающих заданному пользователем условию
4. Алгоритм нахождения минимального (максимального) числа произвольной последовательности чисел
5. Алгоритм поиска всех делителей произвольного целого положительного числа
6. Алгоритм определения, является ли заданное пользователем целое положительное число простым
7. Алгоритм подсчёта суммы, количества и произведения цифр произвольного целого положительного числа
8. Алгоритм поиска НОД и НОК

Дополнительно проверяемые элементы

Помимо основных универсальных алгоритмов на экзамене проверяется умение составлять и применять различные алгоритмы обработки последовательностей, обработки цифр числа, форматированный вывод и прочие элементы, тематическое содержание которых соответствует описанному выше тематическому содержанию.

В частности, проверяются следующие умения по составлению алгоритмов и выполнению необходимой программной обработки:

1. Алгоритм определения минимального (максимального) из трёх чисел без использования циклов
2. Составление логического выражения, отвечающего заданной области на двумерной плоскости
3. Вывод в табличном виде (с использованием форматного вывода) с заданным шагом (значений функции, стоимости товара по граммам и так далее)
4. Вывод по заданному шаблону
   Пример: вводится натуральное N. Вывести треугольник чисел.
   Для N=3 на экран выводится:
   1
   22
   333

5. Вывод последовательности чисел по заданному условию.
   Пример: вывести квадраты 100 нечётных чисел
6. Нахождение и вывод заданного количества первых чисел по заданному условию.
   Пример: найти первые 100 чисел, делящихся на 4, 5 и 6 одновременно
7. Обработка последовательности с заданным или незаданным размером (количеством элементов последовательности).
8. Обработка ряда чисел с заданным количеством элементов (нахождение суммы ряда, вывод квадратов и так далее)
9. Выделение любой цифры N-значного числа, если заранее известно значение N (без использования циклов). Нахождение суммы всех цифр
10. Подбор параметров по условию.
    Пример: найти все пары чисел, произведение которых даёт заданное целое число
11. Вывод всех N-значных чисел, отвечающих заданному условию.

    Пример условия: число содержит ровно один «0»

Контрольные вопросы для самопроверки

Для самоконтроля изученного тематического содержания предлагаются следующие контрольные вопросы для самопроверки:

1. Что такое алгоритм? Какие бывают виды алгоритмических структур?
2. Какие способы записи алгоритма вы знаете?
3. Что такое диалоговая форма программы и как она организуется в языке Python?
4. Какие параметры есть у команды «print()»?
5. Какие типы данных вы знаете и для чего они используются?
   Назовите отдельные операции, применимые лишь к определённым типам данных
6. Что такое операторы целочисленного деления и как они используются в решении различных задач? Привести конкретные примеры
7. Что такое операторы целочисленного деления? Чем целочисленное деление отличается от простого арифметического?
   Как использовать операторы целочисленного деления для определения чётности числа и выделения его цифр?
8. Для чего используется условная конструкция? Какие бывают формы условных конструкций языка Python? Запишите общий вид различных форм условных конструкций

9. Что такое логическое высказывание? Что такое логические операции? Какие основные логические операции вы знаете?
   Продемонстрируйте применение этих операций на конкретных примерах, приведите составное логическое выражение
10. Как задаётся и для чего применяется логический тип данных? Какие значения принимают переменные этого типа данных?
    Приведите примеры применения логического типа данных
11. Назначение циклической алгоритмической структуры. Общие принципы построения и структурные составляющие любого цикла.
12. Операторы циклов языка Python. Их назначение, отличия друг от друга и особенности использования в решении различных задач
13. Оператор «**FOR**». Назначение, структура, способы задания и особенности использования в решении различных задач
14. Оператор «**WHILE**». Назначение, структура, способы задания и особенности использования в решении различных задач
15. Отличия построения и особенностей использования операторов «**WHILE**» и «**FOR**» языка Python
16. Для чего используются и как задаются вложенные циклы? Особенности построения и выполнения вложенных циклов.
17. Приведите пример задания вложенного цикла

Структура зачётного билета

Зачётный билет состоит из 6-ти заданий, некоторых из которых могут состоять из нескольких частей.

Первое задание

Первое задание проверяет основные базовые теоретико-практические знания и навыки учащегося и представлено несколькими небольшими тестовыми вопросами и тестовыми заданиями на умение строить и читать различные циклические структуры, использовать операторы целочисленного деления, переходить от одного циклического оператора к другому, понимать количество выполнений циклического алгоритма и понимание других базовых вопросов программирования или особенностей использования языка Python.

Пример задания

1. Что выведет следующий фрагмент программы:

```
a = 50
```

```
a = int(input())

a = a + 25

print('a')
```

2. Сколько раз выполнится цикл:

```
for i in range(5, 16, 2):

    print(i)
```

3. Чему будет равно значение переменной «k» после выполнения следующего выражения:
   `k = 16 % 5`
4. Чему будет равно значение переменной «k» после выполнения следующего выражения:
   `k = 5 % 16`
5. Сколько раз выполнится цикл:

```
n = 15
while n >= 5:
    n = n - 3
```

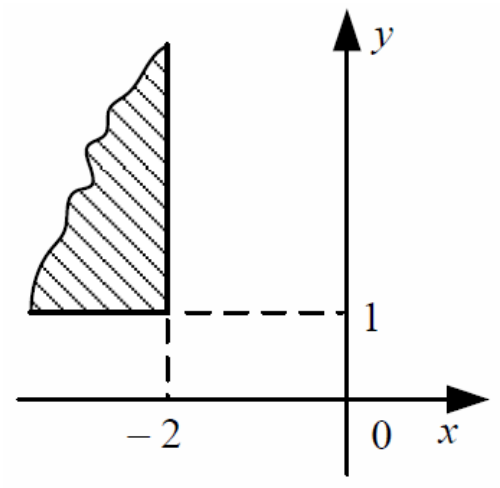
Второе задание

Второе задание проверяет знание и умение использовать элементарные логические операции «**ИЛИ**», «**И**», «**НЕ**», записывая на их основе составные логические выражения для определения принадлежности заданной пользователем точки той или иной геометрической области (областям), определённой (определённых) в задании. Области могут быть ограничены, представляя собой очерченную фигуру, например, фигуру прямоугольника, могут иметь бесконечную протяжённость в одном или нескольких направлениях.

Примеры задания

Пример 1

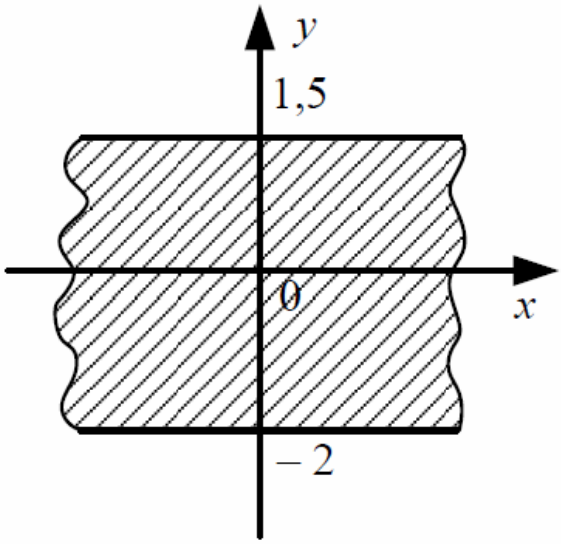
Напишите программу, определяющую, попадает ли указанная пользователем точка с координатами (**x;y**) в заштрихованную область:



При задании логического условия использование двойных неравенств недопустимо.

Пример 2

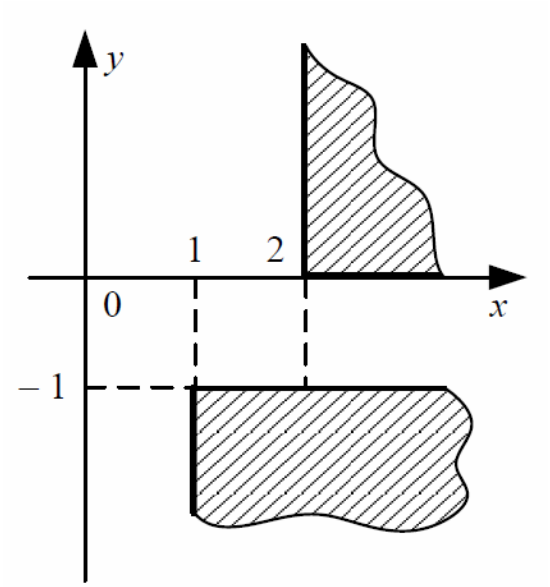
Напишите программу, определяющую, попадает ли указанная пользователем точка с координатами (**x;y**) в заштрихованную область:



При задании логического условия использование двойных неравенств недопустимо.

Пример 3

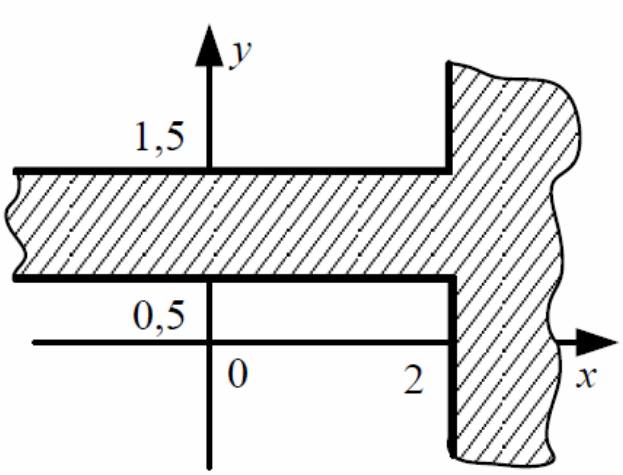
Напишите программу, определяющую, попадает ли указанная пользователем точка с координатами (**x;y**) в какую-либо заштрихованную область:



При задании логического условия использование двойных неравенств недопустимо.

Пример 4

Напишите программу, определяющую, попадает ли указанная пользователем точка с координатами (**x;y**) в заштрихованную область:



При задании логического условия использование двойных неравенств недопустимо.

Третье задание

Третье задание проверяет умение читать, составлять и применять элементарные математические алгоритмы для решения односложных задач на примере применения различных циклических операторов и переходов между ними.

Задание выполняется на бумаге без использования средств вычислительной техники.

Третье задание состоит из двух частей:

1. Задача на умение совершать переход между различными циклическими операторами, записывать один и тот циклический алгоритм с использованием разных циклических операторов
2. Создание итерационной таблицы пошагового выполнения циклического алгоритма с записью значений участвующих в цикле переменных и счётчиков цикла. Для удобства выполнения учащимся данной части зачётного задания на экзаменационном листе будет подготовлена итерационная таблица с необходимыми переменными и пустыми ячейками для самостоятельного заполнения.

Примеры задач первой части

Пример 1

1. Перепишите записанный ниже фрагмент программы с использованием оператора «**FOR**»:

```
x = -100
while x <= 12:
    kvadrat = x * x
    print(kvadrat)
    x = x + 10
```

Пример 2

2. Перепишите записанный ниже фрагмент программы с использованием оператора «**WHILE**»:

```
for x in range(-100, 12, 10):
    kvadrat = x * x
    print(kvadrat)
```

Примеры задач второй части

Заполните итерационную таблицу пошагового выполнения фрагмента циклического алгоритма, представленного ниже:

Пример 1

```
x = 7
while x > -4:
  k = x * x
  x = x - 2
```

№ итерации	Переменная «k»	Переменная «x»
0 – начальная инициализация	-	7
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Пример 2

```
x = 3
k = 1
for i in range(5):
  x = x + 1
  k = k * x
```

№ итерации	Переменная «i» – счётчик цикла	Переменная «x»	Переменная «k»
0 – начальная инициализация	-	3	1
1			
2			
3			
4			
5			
6			
7			

8			
9			
10			

Четвёртое задание

Четвёртое задание проверяет элементарные навыки программирования, умение составлять алгоритм решения задачи, записать и реализовать его на языке программирования, отладить программу и в конечном счёте получить полностью работоспособный программный код, правильно работающий на всех входных тестовых данных, то есть выдающих верные выходные данные при любых входных данных, включая все ограничения из условия задачи.

Примеры заданий

Пример 1

Непустая последовательность целых чисел, вводимых с клавиатуры, заканчивается нулём. Найдите произведение нечётных чисел.

Пример 2

Найдите значение выражения, если количество слагаемых определяет пользователь программы:

1 + ½ + 1/3 + ¼ + … + 1/N, где N – номер соответствующего члена последовательности

Пример 3

Пользователь задаёт размер и саму последовательность.

Определите, сколько раз в данной последовательности встретилось максимальное и минимальное число.

Пример 4

Определите, является ли заданная пользователем последовательность убывающей?

Пользователь вводит последовательность до значения «0», которое является признаком конца ввода и не входит в саму последовательность.

Пятое задание

Пятое задание представлено задачами, в решении которых обычно используются вложенные циклы. Подобный тип заданий не обладает большой

математической и алгоритмической сложностью, позволяя экзаменаторам оценить основные навыки программирования учащегося.

Для решения этого задания необходимо хорошо понимать принципы композиции алгоритма, особенности выполнения всех видов алгоритмических структур, в частности, циклических структур разного вида.

Примеры заданий

Пример 1

Пользователь задаёт два целых числа N и K, определяющих диапазон целых чисел [N, K].

Найдите в этом диапазоне число с максимальной суммой делителей.

Пример 2

Пользователь задаёт два целых числа N и K, определяющих диапазон целых чисел [N, K].

Определите общее количество простых делителей у всех чисел из этого диапазона. Если у одного числа два и более простых одинаковых делителя (как, например, у числа «25» это два делителя «5»), считать только один из них.

Пример 3

Пользователь программы вводит два целых положительных числа «**a**» и «**b**», задающих диапазон целых чисел **[a; b]**.

Найдите в пределах этого диапазона и выведите на экран все числа, начинающиеся и заканчивающиеся на цифру «8».

Шестое задание

В шестом задании учащимся необходимо найти и исправить все ошибки, допущенные в программном коде при решении поставленной задачи.

Учащийся анализирует полное условие задачи, что дано и что нужно сделать, вникает в представленное решение задачи, после чего он должен определить, в каком месте или местах программного кода допущены ошибки, приводящие к неверному решению задачи. В данном задании потребуется не только указать на ошибки и обосновать их, но и предложить вариант их исправления для верного решения поставленной в условии задачи.

Данное задание проверяет умение анализировать чужой программный код, проводить его итерационное исполнение в уме или на бумаге, следить за изменением переменных при произведении над ними математических операций и на основании собранных данных находить ошибки в представленном коде программы.

Примеры задания

Пример 1

Дано целое положительное число N. Необходимо определить максимальное значение степени числа 2, на которое N делится без остатка. Например, для N = 2016 нужно получить результат 32, а для N = 2017 – результат 1.

Для решения этой задачи ученик написал программу, но, к сожалению, его программа неправильная:

```
n = int(input())
k = 0
while n % 2 == 0:
    n = n//2
    k = k + 2
print(k)
```

Последовательно выполните следующее:

1. Напишите, что выведет эта программа при вводе N = 2016?
2. Найдите в программе все ошибки (их может быть одна или несколько). Для каждой ошибки выпишите строку, в которой она допущена, и приведите эту же строку в исправленном виде

Обратите внимание: вам нужно исправить приведённую программу, а не написать свою. Вы можете только заменять ошибочные строки, но не можете удалять строки или добавлять новые. Заменять следует только ошибочные строки.

Пример 2

На обработку поступает натуральное число, не превышающее 10^9. Нужно написать программу, которая выводит на экран минимальную чётную цифру этого числа. Если в числе нет чётных цифр, требуется на экран вывести «NO».

Для решения этой задачи ученик написал программу, но, к сожалению, его программа неправильная:

```
N = int(input())
minDigit = N % 10
while N > 0:
  digit = N % 10
  if digit % 2 == 0:
    if digit < minDigit:
      minDigit = digit
  N = N // 10
if minDigit == 0:
  print("NO")
else:
  print(minDigit)
```

Последовательно выполните следующее:

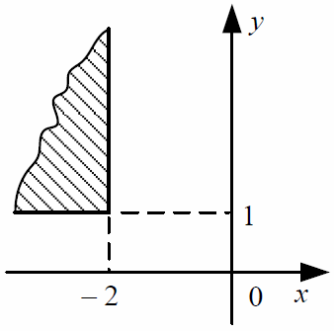
1. Напишите, что выведет эта программа при вводе N = 231?
2. Найдите в программе все ошибки (их может быть одна или несколько). Для каждой ошибки выпишите строку, в которой она допущена, и приведите эту же строку в исправленном виде

Обратите внимание: вам нужно исправить приведённую программу, а не написать свою. Вы можете только заменять ошибочные строки, но не можете удалять строки или добавлять новые. Заменять следует только ошибочные строки.

Образец зачётного билета

Билет № ___

Фамилия и имя, класс:

№ задания	Задание / Ссылка на задание	Ответ	№ задачи в задании
1	Что выведет следующий фрагмент программы: **a = 50** **a = a + 25** **print('a')**		
	Сколько раз выполнится цикл: **for i in range(18, 32, 2):** **print(i)**		
	Чему будет равно значение переменной «k» после выполнения следующего выражения: **k = 12 % 5**		
	Чему будет равно значение переменной «k» после выполнения следующего выражения: **k = 8 % 10**		
	Сколько раз выполнится цикл: **n = 19** **while n >= 6:** **n = n - 3**		
2	Напишите программу, определяющую, попадает ли указанная пользователем точка с координатами (x;y) в заштрихованную область:  При задании логического условия использование двойных неравенств недопустимо.		

3.1	Перепишите записанный ниже фрагмент программы с использованием оператора «**while**»: **a = 3** **for i in range(-5, 3):** **a = a * 2**	
3.2	Составьте итерационную таблицу пошагового выполнения фрагмента циклического алгоритма, представленного ниже: **x = 7** **while x > -4:** **k = x * x** **x = x – 2** *Для выполнения данного задания необходимо заполнить уже подготовленную таблицу, которую выдаёт экзаменатор вместе с экзаменационным билетом*	
4	https://official.contest.yandex.ru/contest/27336/enter	Задача А
5	https://official.contest.yandex.ru/contest/27337/enter	Задача А

6	Дано целое положительное число N. Необходимо определить максимальное значение степени числа 2, на которое N делится без остатка. Например, для N = 2016 нужно получить результат 32, а для N = 2017 – результат 1.
	Для решения этой задачи ученик написал программу, но, к сожалению, его программа неправильная:
	`n = int(input())`
	`k = 0`
	`while n % 2 == 0:`
	`n = n // 2`
	`k = k + 2`
	`print(k)`
	
	Последовательно выполните следующее:
	1. Напишите, что выведет эта программа при вводе N = 2016?
	2. Найдите в программе все ошибки (их может быть одна или несколько). Для каждой ошибки выпишите строку, в которой она допущена, и приведите эту же строку в исправленном виде
	Обратите внимание: вам нужно исправить приведённую программу, а не написать свою. Вы можете только заменять ошибочные строки, но не можете удалять строки или добавлять новые. Заменять следует только ошибочные строки.	

Список рекомендуемой литературы

1. Д.М. Златопольский «Основы программирования на языке Python»
2. Зед Шоу «Лёгкий способ выучить Pyhton 3»
3. Майк Макграт «Python. Программирование для начинающих»
4. Эл Свейгарт «Автоматизация рутиных задач с помощью Python: практическое руководство для начинающих»
5. Задачники:
    a. Павловская Т.А. "Программирование на языке высокого уровня"
    b. Т.Ю. Грацианова «Программирование в примерах и задачах»
    c. Д.М. Златопольский «Сборник задач по программированию»
    d. В.А. Дагене, Г.К. Григас, К.Ф. Аугутис «100 задач по программированию»

Для подготовки подойдут любые номера изданий и сами издательства.