Задание 24

https://yandex.ru/tutor/subject/problem/?problem_id=T4916

Дано целое положительное число N>=10. Необходимо найти наименьшую сумму двух соседних цифр в десятичной записи N. Например, для N=2018 нужно получить ответ 1, а для N=2030 ответ 2.

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

Ниже эта программа для Вашего удобства приведена на пяти языках программирования (мы рассмотрим версию на Питоне).

```
n = int(input())
m = 10
while n >= 10:
    d1 = n // 10
    d2 = n // 10 % 10
    s = d1 + d2
    if s < m:
        m = s
    n //= 10
print(m)
```

Последовательно выполните следующее.

1. Напишите, что выведет эта программа при вводе N=2018.
2. Приведите пример числа N, при котором программа выведет верный ответ. Укажите этот ответ.
3. Найдите в программе все ошибки (известно, что их не больше двух) и исправьте их. Для каждой ошибки выпишите строку, в которой она допущена, и приведите эту же строку в исправленном виде.

Достаточно указать ошибки и способ их исправления для одного языка программирования.

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, из-за которых программа может выдать неверный ответ: за исправления, внесённые в любые другие строки, баллы будут снижаться.

Решение

Чтобы решать такого рода задачи (а условия задач типовые) вы должны понимать арифметические операции целочисленного деления, вычисления остатка от целочисленного деления, работу циклов и пошагово их раскладывать.

Чтобы успешно решить данную задачу нужно потренироваться (решить 5-7 аналогичных задач) и наработать систему размышлений. Сначала поймите, что именно нужно было делать и только потом расшифровывайте чужое решение.

Приступаем к разбору.

Итак, в задаче нужно «*найти **наименьшую** сумму двух соседних цифр*» в числе, следовательно, нужно будет делать перебор. Запустим **цикл**, на каждой итерации цикла будем **отделять пару цифр**, находить их **сумму** и **сравнивать** не стала ли она меньше, чем была ранее. Когда цифры в числе **закончатся**, выйдем из цикла и напечатаем минимальную сумму. Значимые места алгоритма я выделил.

Принципиальным моментом будет являться способ отделения цифр – обычным способом является последовательное отделение цифр с правого края операцией **%10** (или **mod 10** для Pascal) до тех пор, пока все цифры числа не закончатся…

Отступление.

Для более глубокого понимания изучите алгоритм поиска суммы цифр целого числа 0<=N<=10000:

```
# алгоритм поиска суммы цифр числа
n = 19340 # тут вводим число
r = 0
while n > 0:
    r += n % 10 # отделить последнюю цифру
    n //= 10 # удалить последнюю цифру
print(r)
```

В данном случае на каждом шаге цикла мы узнаём последнюю цифру и добавляем её к переменной r, к которой последовательно будут добавлены значения всех цифр. Кроме того, на каждом шаге переменная n делится нацело на 10, тем самым мы «избавляемся от последней цифры», например, 128//10=12, то есть, если было 128, то стало 12 – и это сокращённое число передаём на следующую итерацию цикла. Цикл продолжается пока число n больше 0.

Вернёмся к нашей задаче – нам нужно за раз отделять по две рядом стоящие цифры.

Чтобы от числа отделить крайнюю справа цифру нужно вычислить остаток от деления на 10, например, так:

128 % 10 = 8 – получаем первую справа цифру числа.

Чтобы взять вторую с конца цифру нужно сначала число поделить нацело на 10, а затем, как и ранее, взять остаток от деления на 10, то есть выполнить два шага:

1) 128 // 10 = 12
2) 12 % 10 = 2

Что можно записать в одну строку:

128 // 10 % 10 – это соответствует переменной **d2** в программе, то есть **d2** это вторая справа цифра в любом числе равном или большем 10.

А вот первую справа цифру в программе, как оказалось, получают неправильно – вот, что написано в программе:

d1 = n // 10, а вот как должно быть:

d1 = n % 10

Получается, что мы уже нашли одну из двух ошибок.

Посмотрим, что дальше происходит в теле цикла:

```
s = d1 + d2
if s < m:
    m = s
n //= 10
```

Сумма двух соседних цифр находится корректно, как мы и обсуждали ранее: `s = d1 + d2`. В случае, если она окажется меньше, чем ранее найденная сумма, то её следует обновить: `m = s`. Перед переходом к следующей итерации цикла следует число сократить в 10 раз нацело, то есть отрезать последнюю цифру: `n //= 10` – и здесь всё корректно.

Ищем вторую ошибку далее. Может условие выхода из цикла некорректно указано: `while n >= 10`. Почему тут сказано, что делай цикл пока n больше или равно 10? Это потому что в теле цикла мы будем из числа брать сразу две цифры – крайнюю справа и вторую справа, то есть мы можем заходить в цикл только, если число имеет хотя бы два разряда (должно быть не менее 10). Значит и тут всё нормально.

Тогда вернёмся к началу программы… Что там есть? А там есть ввод числа – написано корректно и инициализация переменной `m = 10`. Почему 10 и что это за переменная такая? Эта переменная отвечает за хранение минимума суммы двух цифр. Давайте обсудим какое значение в принципе могло бы там оказаться: если цифры 1 и 1, то 2, если цифры 9 и 9, то 18, то есть в m может быть записано от 2 до 18. А нам изначально туда записали 10 – это ошибка. Например, если бы в число n ввели 99, то найденная сумма цифр **s** была бы равна 18, но она бы не обновилась в переменной m, так как не сработало бы условие `if s < m`, ведь в m изначально поставили 10. Это и есть вторая ошибка, чтобы программа работала корректно, нужно инициализировать m числом 18.

Вот корректная версия программы:

```
n = int(input())
m = 18
while n >= 10:
    d1 = n % 10 # отделяем крайнюю цифру
    d2 = n // 10 % 10 # отделяем вторую справа цифру
    s = d1 + d2 # находим их сумму
    if s < m: # если сумма меньше, чем ранее
        m = s # обновляем минимальную сумму
    n //= 10 # отрезаем последнюю цифру
print(m)
```

Исправленные две строчки показаны красным цветом – это был ответ на третий пункт задания.

Теперь разберём поиск ответа на второй пункт: «Приведите пример числа N, при котором программа выведет верный ответ. Укажите этот ответ».

Можете просто подобрать ответ, но можно и подумать…

В нашем разборе мы отметили ранее, что переменная m была неправильно инициализирована числом 10 – это значит, что условие **`if s < m`** будет корректно работать только для суммы цифр меньше или равной 10, например, для числа 45 (по условию задачи, число не может быть меньше 10). Но у нас в исходной программе есть ещё одна ошибка при нахождении крайней правой цифры – в программе было написано **d1 = n // 10**, а должно быть так **d1 = n % 10**. Когда эти две строки дадут правильный ответ, когда это будет двузначное число из одинаковых цифр, например, **55** (или 44, 33, 22, 11). Это и есть ответ на второй вопрос.

Осталось только найти ответ на первый – что будет при вводе числа 2018?

Чтобы не сделать ошибку, постройте рассуждения в виде таблицы пошагового изменения всех переменных:

	до цикла			10	2018
№ шага	d1	d2	s	m	n
1	201	1	202	10	201
2	20	0	20	10	20
3	2	2	4	4	2

В четвёртый шаг цикла мы не заходим, так как на исходе третьего шага переменная n стала равна 4, что не соответствует условию цикла.

На экран выводится значение минимума этой некорректной программы – значение переменной m = **4** – это ответ на первый вопрос.

Собственно, всё, в этой задаче нужен небольшой набор знаний, но много работы и внимательности – нарабатывается практикой. Кроме того, в рекомендациях по решению ЕГЭ по информатике указано, что из 4-х часов на первую часть желательно отвести 90 минут, а на вторую оставить 2,5 часа. То есть у вас на 24, 25, 26 и 27 задачи 150 минут!.. Так что особенно не спешите, делайте все размеренно, по шагам, чтобы не было невынужденных ошибок.