Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет геосистем и технологий»
(СГУГиТ)

В.Е. Терещенко

ПРОГРАММИРОВАНИЕ ДЛЯ РЕШЕНИЯ ГЕОДЕЗИЧЕСКИХ И МАРКШЕЙДЕРСКИХ ЗАДАЧ

Лабораторная работ №2
Условия. Множественные условия. Условные операторы

Новосибирск
СГУГиТ
2024

ЛАБОРАТОРНАЯ РАБОТА №2

Условия

Условия всегда, не только в языках программирования, но и в обычных разговорных языках, связаны со словом «если». На англ. язык слово «если» переводится как «**if**». Python воспринимает это слово как команду. PyCharm как и другие функции подсвечивает ее цветом. Например, нужно создать программу, которая бы называла студентов отличниками, хорошистами и троечниками, в зависимости от полученной ими оценки. Введите следующий код. (grade – от. англ. – «оценка»):

```
grade = 5
if grade == 5:
    print(‘Студент отличник!’)
```

В Python существует набор правил ввода кода, которые обязательно должны быть соблюдены, иначе будет возникать ошибка. Строка, содержащая команду «**if**» обязательно должна быть закончена символом двоеточия «**:**». После задания условия следующая строка обязательно должна начинаться с отступа (indent – с англ. «отступ», «абзац»), то есть с символа табуляции или четырех пробелов. Согласно некоторым источникам (см. курс лекций), лучше всегда использовать четыре пробела, вместо кнопки tab. Это правило навсегда исключит путаницу и некоторые возможные ошибки. Отступы играют очень важную роль. Они сообщают Python что, если условие верно, нужно выполнять все действия, которые отмечены табуляцией. Если условие не верно, то игнорировать все действия, отмеченные табуляцией, например:

```
grade = 4
if grade == 5:
    print(‘Студент отличник!’)
print(‘Программа завершена!’)
```

То есть та часть кода, которая относится к функции «**if**», отделенная табуляцией, выполняться не будет. При этом программа продолжит выполнение, но уже той части кода, которая не относится к функции «**if**», а которая введена далее и без табуляции (отступов) – print("Программа завершена!").

Обратите внимание на двойной равно (символ «==»). Двойное равно применяется всегда при сравнении значений, в том числе при работе с функцией «**if**».

Функция «**if**» выполняет проверку условия. Если проверка условия дает значение **True** (правда), то дальнейшее выполнение кода, связанного с функцией «**if**» будет выполняться. Если проверка условия дает значение **False** (ложь), то весь блок кода, относящийся к команде «**if**» выполняться не будет.

Продолжим дописывать программу. Нужно ввести еще несколько условий, чтобы теперь отличать от отличников хорошистов и троечников. Введите:

```
grade = 5
if grade == 5:
    print(‘Студент отличник!’)
if grade == 4:
    print(‘Студент хорошист!’)
if grade == 3:
    print(‘Студент троечник!’)
print(‘Программа завершена!’)
```

Теперь в зависимости от того, какая оценка присвоена переменной – будет выведена характеристика студента.

Существует еще две команды, связанные с работой с условиями. Первая – команда else (с англ. можно перевести как «иначе», «по-другому»). Вторая – это elif (с англ. производная от else if – иначе если). Эти команды помогают конкретизировать условия и упростить синтаксис кода, а иногда без них обойтись вообще нельзя. Предположим, что программист может ошибиться и случайно вместо отметки «3», «4» или «5» введет случайное число или «зачетно» или «не зачтено». В этом случае программа должна будет вывести сообщение об ошибке. Название переменной «grade_geodesy» можно перевести как «оценка по предмету Геодезия». Пример:

```
grade_geodesy = "зачтено"
if grade_geodesy == 5:
    print("Студент отличник!")
elif grade_geodesy == 4:
    print("Студент хорошист!")
```

```
elif grade_geodesy == 3:
    print("Студент троечник!")
else:
    print("Ошибка ввода данных")
```

То есть, если переменная grade_geodesy будет равна не «5», не «4» и не «3», будет выведено сообщение, что «Ошибка ввода данных».

Задание 11

1. Напишите программу, введя переменную grade_main_surveyor (оценка по предмету маркшейдерия). Задайте ей значение «зачтено» или «не зачтено». Установите условия: если переменная принимает значение «зачтено», то должен быть вывод «студент отличник», если переменная принимает значение «не зачтено», то студент «двоечник», если введено отличное от этих двух значение переменной, то программа должна вывести на экран текст «ошибка ввода данных».
2. Результат представьте в виде скопированного кода в отчет в word-документе.
3. Приведите скриншот окна PyCharm с результатом выполненной программы в отчет.

Множественные (вложенные) условия

Ту же самую задачу по получению допуска к экзамену для студента можно решить с помощью множественных условий. В языке Python это делается с помощью отступов. Как в обычных условиях после команды «if» на следующей строке ставится табуляция или четыре пробела, так и во множественных условиях это работает, даже если четыре пробела уже использовались ранее в этом же условии. Только в этом случае необходимо поставить уже 8 пробелов. Пример кода:

```
abstract = 'сдан'
test = 5

if abstract == 'сдан':
    if test == 3:
        print('Допуск к экзамену есть')
    elif test == 4:
        print('Допуск к экзамену есть')
    elif test == 5:
        print('Допуск к экзамену есть')
    elif test == 2:
        print('Допуск не получен')
    else:
        print('Ошибка ввода данных')
else:
    print('Допуск не получен')
```

Но в этом коде есть одна неучтенная деталь. Если в значении переменной «abstract» будет введено вместо «сдан», например слово «слан», допущена случайная опечатка, результат будет «'Допуск не получен'», хотя реферат сдан.

Задание 12

1. Исправьте код так, чтобы случайная ошибка в значении переменной «abstract» не приводила к неадекватному выводу результата, а программа выводила текст-предупреждение о том, что «введенная информация вероятно ошибочна».
2. Добавьте еще одно вложенное условие в программу. Можете придумать свое, или, к примеру, чтобы получить допуск к экзамену, помимо сданного реферата и удовлетворительной и выше оценки за тест, необходимо сдать дополнительно словарь русско-английских терминов по данной дисциплине. Условия те же, «сдан» и «не сдан». Слово «словарь переводится на англ. как «dictionary».
3. Результат представьте в виде скопированного кода в отчет в word-документе.
4. Приведите скриншот окна PyCharm с результатом выполненной программы в отчет.

Задание 13

1. Напишите программу согласно следующим требованиям. У студента две дисциплины геодезия и маркшейдерия, соответственно им есть оценки, которые передадим в переменные: grade_geodesy, grade_main_surveyor. По геодезии ставится отметка по пятибалльной шкале, а по маркшейдерии ставится «зачтено»

или «не зачтено». По результатам оценок за эти два предмета необходимо дать характеристику студенту. Для этого потребуются вложенные условия, условные операторы. Если оценка по геодезии тройка и выше и по маркшейдерии «зачтено», то студент троечник, хорошист или отличник. Но если маркшейдерия «не зачтено» или есть двойка по геодезии, тогда нужно вывести что «студент двоечник». Необходимо, чтобы были предусмотрены все варианты. Если вдруг будет ошибка ввода данных, ошибка в слове «зачтено» или по геодезии будет стоять оценка, например, 6 или 1, необходимо, чтобы программа выдала сообщение «ошибка ввода данных».

2. Результат представьте в виде скопированного кода в отчет в word-документе.
3. Приведите скриншот окна PyCharm с результатом выполненной программы в отчет.

Условные операторы

В языке Python существуют такие операторы как and, or, not. Они используются для облегчения ввода кода, а иногда без них обойтись не удается вовсе. Рассмотрим пример. Нужно написать программу, которая допускает студентов до экзамена. Условия допуска следующее, написать тест на оценку выше двойки и сдать реферат.

```
abstract = 'сдан'
test = 5
if abstract == 'сдан' and (test == 3 or test == 4 or test == 5):
    print('Допуск к экзамену есть')
else:
    print('Допуск не получен')
```

Оператор not используется в случае, когда необходимо инвертировать поведение функции if. Например, чтобы не прописывать в коде длинную строку со множеством or (test == 3 or test == 4 or test == 5), зная, что оценка может иметь значение от 2 до 5, можно упростить код, написав:

```
abstract = 'сдан'
test = 5
if abstract == 'сдан' and not test == 2:
    print('Допуск к экзамену есть')
else:
    print('Допуск не получен')
```

получим абсолютно тот же результат. Однако стоит понимать, что значение оценки за тест может быть выставлено ошибочно, например 1 или 6, но это уже другой случай. Здесь приведен лишь пример использования оператора.

В Python существуют правила приоритетности выполнения действий. Они всегда соблюдаются несмотря на то, что код всегда выполняется слева-направо и сверху-вниз. В табл. 4 приведена приоритетность выполнения действия в Python. Самые приоритетные операции вверху, а снизу – с низким приоритетом. Вычисления выполняются слева направо, то есть, если в выражении встретятся операторы одинаковых приоритетов, первым будет выполнен тот, что слева. Оператор возведения в степень исключение из этого правила. Из двух операторов ** сначала выполнится правый, а потом левый.

Таблица 4 – приоритетность выполнения операторов в Python

Операторы	**Описание**
()	Скобки
**	Возведение в степень
+x, -x, ~x	Унарные плюс и минус
*, /, //, %	Умножение, деление, целочисленное деление, остаток от деления
+, -	Сложение и вычитание
<<, >>	Битовые сдвиги
&	Битовое И
^	Битовое исключающее ИЛИ (XOR)
|	Битовое ИЛИ
==, !=, >, >=, <, <=, is, is not, in, not in	Сравнение, проверка идентичности, проверка вхождения
not	Логическое НЕ
and	Логическое И
or	Логическое ИЛИ

Исходя из этих правил при написании кода необходимо использовать скобки для адекватной работы программы.

Задание 14

1. Перепишите код из задания 12 с переменными «test» и «abstract» так, чтобы в нем были использованы операторы and, or, not и код адекватно работал.
2. Проанализируйте, насколько сократилось количество строчек кода и насколько код стал более прост в прочтении.
3. Результат представьте в виде скопированного кода в отчет в word-документе.
4. Приведите скриншот окна PyCharm с результатом выполненной программы в отчет.

Задание 15

1. Перепишите код из задания 13 с переменными «grade_geodesy» и «grade_main_surveyor» так, чтобы в нем были использованы операторы and, or, not и код адекватно работал.
2. Результат представьте в виде скопированного кода в отчет в word-документе.
3. Приведите скриншот окна PyCharm с результатом выполненной программы в отчет.