|  |  |  |
| --- | --- | --- |
| **Python . Control lesson number 1** | | |
| **Student:** | **Lessons Completed:** | **Date:** |
| **Profile link :** | | |

|  |  |  |
| --- | --- | --- |
| **Theory**(oral survey, 1 -2 question from the chapter, in case of difficulty, you can ask another question from the chapter) | | Mark  +/- |
| **Topic 1 Variables**(Covered in lessons: 0 , 1.1 ) | |  |
| What are variables ? | The answer fits the pattern: " This is something that can be named somehow and some values can be put ​​there. " |  |
| How do I create a variable? | Write the name of the variable and assign / put / write down some values ​​there. |  |
| How to assign / put / write some values ​​to a variable ? | Use equal sign . |  |
| What names can be used for variables, or can there be a space in variable names? | Any name, but you cannot put a space. |  |
| How do I increase the value of a variable by 1? | Write to a variable the sum of the old value of the variable plus 1, or, using the example of a variable, said everything in the words < variable name > = < variable name > +1 |  |
| **Topic 2 Data Types**(Covered in lessons: 1.1 ) | |  |
| What are the types of data? | There are strings, numbers, and booleans (logical, true-false). |  |
| If we display a comparison of two numbers, what do we see? What is it? | True or False . |  |
| Can we add different types of data, if so, how ? | No, or use the conversion str ( ) or int () . |  |
| What are the str ( ) and int () commands for ? | Makes a string out of anything, makes a number out of anything . |  |
| **Topic 3 Conditional statements**(1.2) | |  |
| What is If? | The language command for the condition . |  |
| Why is if needed ? | To check conditions.  In order for some lines to work occasionally. |  |
| What is a condition? give an example | If A is greater than B ... (or any other example of inequalities or logical expressions) . |  |
| Can we check two conditions at once? | Yes . |  |
| What are **or** and **and** for ? | To combine terms. |  |
| How is **or** different from **and** ? | The code will be executed when at least one of the conditions is correct. The code will be executed when two conditions are true. (Full compliance with the answer is not necessary, it is enough that this thought is present in the student's answer) |  |
| **Topic 4 Cycles**(Covered in lessons: 1, 1.3) | |  |
| What are loops/cycles? | Commands that repeat other commands . |  |
| What are the types of cycles? | **While** and **for** |  |
| How many times does **while** and **for** work . | **While** works as long as the condition is true, **for** works as many times as it was told. |  |
| How do I make a **for** that will work 5 times? | Use **range(5)** . |  |
| Can **for** be infinite ? | No . |  |
| How do I make an infinite**while**? | Write **while True**. |  |
| Which command stops the loop? | **Break** command . |  |
| **Total** | |  |

|  |  |  |
| --- | --- | --- |
| **Practice**.  Note :   * mark (2) is put if the student independently invented and implemented the solution, * mark (1) is set if the solution was invented or implemented with a hin , or a hint was provided when the error was eliminated * the mark (0) is set if the solution was invented and implemented with a hint, or a hint was provided when fixing an error | | Mark  0..2 |
| **Part 1 (mandatory)** | |  |
| The obtaining of a number from the user was implemented | Created a variable and wrote a value from the keyboard |  |
| The receiving a bet from a user was implemented | Created a variable and wrote a value from the keyboard |  |
| Type conversion was completed | You can cast types, both conditions together, and when getting values ​​from the keyboard |  |
| The **random** library was included | Or any other way to get a random number |  |
| A dice roll was implemented | By condition, you need to get a random number twice |  |
| The sum of the dropped points to a variable was saved | Created a separate variable and wrote the sum to it |  |
| implemented the first condition correctly and by himself | was able to transfer the first condition by himself from the task to the code |  |
| All the conditions from the problem were joined together | The conditions are described in the problem, we connect the conditions using **elif**, if all the conditions connected are true, then this the submission of the second part |  |
| The winning and losing bets was realized | If you win, the bet should be added to the total number of points, if you lose, it should be deducted. |  |
| The condition from the problem was compiled using the correct operator | In the problem, the conditions are linked by the union “ and ” in the code, the operator and |  |
| The exit from the game at the request of the player was implemented | Just as in the homework to the lesson 1.3 |  |
| Exit box was placed in the right place | If the offer to quit comes between the die roll and the block condition, or in any other inappropriate place, it will not count. |  |
| Added a variable to store the total number of points | " Variable\_name " = 100 |  |
| The creation of the variable for the total number of points was placed in the correct place | Create variable in loop wrong |  |
| The game does not end after the first move | All code except for connecting the library and creating a variable (total number of points) is written inside **while** |  |
| The exit from the game was implemented when spending all points | It can be implemented as a condition for the loop, as well as creating a boolean variable whose value will change inside the loop. |  |
| **Part 2 (optional)**  Note, for performing I of this part, you can specify existing drawbacks | |  |
| The user number range check was added | "Calls any number between 2 and 12" |  |
| Protected against foul play | In the game, you can make a negative bet and the bet is more than the number of points than the player has. It's not fair |  |
| Got rid of redundant command repetition | The dice roll can be done in one line or in a cycle, as well as remove unnecessary convertions, if any. |  |
| Compiled a completely correct conditions block | All conditions are connected correctly. |  |
| It's clear what's going on in the game | Messages from the game should be displayed on the screen in sufficient volume and with the necessary content. It should be possible to play. |  |
| **Total** | |  |

**Practical notes :**

**Assessment of student's attitude to learning (on a scale of 1 to 5):**

Interest -

Motivation -

Engagement -

Academic achievement –

***During the lesson, the student showed himself as (choose 2-3):***

* Attentive
* Plodding
* Curious
* Hardworking
* Creative, innovative
* Inventive
* Your option

***Qualities requiring further attention and development (choose 2-3 ):***

* Mindfulness,
* Perseverance
* Hard work
* Homework
* Mathematical calculations
* Ability to present, tell and explain logically
* Creation
* Initiative

***Topics of particular interest to the student:***

* Programming
* Painting
* The gameplay itself

***Recommendations from the teacher:***

* Continue course
* Change course
* Increase the number of independent studies to….
* Reduce the number of independent studies to….