Федеральное агенство связи
Федеральное государственное бюджетное образовательное учреждение
высшего образования "Сибирский государственный университет
телекоммуникаций и информатики"

**Кафедра ТСиВС**

# Лабораторная работа №3
# Создание бота для участия в HonorCup

Выполнила:
студенка группы ИА-831
Угольникова Екатерина Алексеевна

Проверил:
ведущий инженер кафедры ТСиВС
Ахпашев Руслан Владимирович

Новосибирск 2020

# Цель работы

Целью данной лабораторной работы является знакомство с языком программирования Python и написание на этом языке алгоритма автоматизированного участия в раундах HonorCup, заполнение базы вопросов и ответов для успешной победы в викторинах.

Основными задачами данной рабораторной работы являются:

1. Написание бота на языке Python;
2. Автоматизация участия в викторине и получения высоких баллов;
3. Взаимодействие с элементами веб-страницы через средства библиотеки Selenium;
4. Работа с форматом данных JSON;
5. Заполнение базы данных вопросов как можно большим количеством правильных ответов;

# Ход работы

Работа бота с базой вопросов организована таким образом, что весь пул вопросов перед началом работы выгружается из JSON файла в оперативную память в виде словаря (QUESTIONS), преобразуется во время участия в викторинах и по завершении работы бота загружается обратно в файл. Но помимо финального сохранения, база вопросов в файл сохряняется в процессе работы бота, чтобы не потерять большое количество данных в случае сбоя программы.

В файле хранится словарь данных следующей структуры:

1. Текст вопроса;
2. Флаг, показывающий известен ли верный ответ на вопрос (0 или 1);
3. Текст верного ответа;
4. Список индексов известных неверных ответов на вопрос (цифры от 0 до 3);
5. Список ответов на вопрос (4 элемента);

Из кода веб страницы с помощью инструментов библиотеки Selenium считывается вопрос раунда (Question) и предложенные ответы (current_answers).

Эти данные поступают на вход функции QualiativeQuestion вместе со словарем QUESTIONS. Функция предполагает полную обработку текущего вопроса.

При первом запуске программы база данных вопросов может быть пуста, поэтому сначала проверяется, пуст ли словарь.

Если словарь пуст, или если в словаре не оказалось конкретного текущего вопроса, то выбирается случайный ответ, проверяется правильный ли он и вопрос добавляется в словарь в необходимом виде.

Если же база вопросов не пуста, то текущий вопрос поочередно сравнивается с каждым вопросом в словаре, пока не будет найдено совпадение.

Когда в базе найден необходимый вопрос, запускается функция обработки этого вопроса QuestionCoincide, которая получает на вход запись найденого в словаре вопроса и список ответов из текущего раунда.

Работу этой функции можно описать следующим алгоритмом:

**Algorithm 1** Когда найдено совпадение вопроса из базы с вопросом из текущего раунда (function QuestionCoincide)

```
1: if флаг правильного == 1 then
2:     найти совпадение правильного с round_answers[j], j ∈ 0..3
3:     выбрать j ответ
4: else
5:     if список индексов неправильных ответов пуст then
6:         сгенерировать случайное число i от 0 до 3
7:         найти совпадение выбранного i ответа (из записи в базе) с round_answers[j], j ∈ 0..3
8:         выбрать j ответ
9:         if ответ j правильный then
10:            поднять флаг правильного
11:            записать j ответ в текст правильного
12:            очистить список ответов на текущий вопрос
13:        else
14:            записать индекс i в список индексов неправильных ответов
15:        end if
16:    else
17:        сгенерировать случайное число i от 0 до 3, исключив возможность выбора чисел, которые есть в списке индексов неправильных ответов
18:        найти совпадение выбранного i ответа (из записи в базе) с round_answers[j], j ∈ 0..3
19:        выбрать j ответ
20:        if ответ j правильный then
21:            поднять флаг правильного
22:            записать j ответ в текст правильного
23:            очистить список индексов неправильных ответов
24:            очистить список ответов на текущий вопрос
25:        else
26:            записать индекс i в список индексов неправильных ответов
27:        end if
28:    end if
29: end if
```

Строки 7-15 и 18-27 объединены в функцию HelpFunction1, получающую на вход запись найденого в словаре вопроса, список ответов из текущего раунда и сгенерированный случайный индекс i.

Помимо функций для обработки вопросов, использованы некоторые вспомогательные функции.

1. waiting() - функция ожидания соперника. Реализована бесконечным циклом до момента, пока на странице не появится вопрос раунда;
2. save_questions() - функция сохранения словаря с вопросами в файл JSON;
3. database_adjustment(database) - функция корректировки базы данных. Функция очищает базу от лишних данных. Если в вопросе указан флаг правильности, то следует очистить список ответов на вопрос и список индексов неправильных ответов. Или если в списке индексов неправильных ответов найдено три элемента (это говорит о том, что 3/4 ответов неправильные, а значит последний оставшийся правильный), то находится этот оставшийся ответ, записывается в текст правильного ответа на вопрос, поднимается флаг правильности и очищаются список ответов и список индексов;

Общий вид работы бота:

1. открытие браузера FireFox;
2. переход на главную страницу викторин;
3. нажатие на кнопку "Сражаться за кубок";
4. выбор раздела викторины (в нашем случае IP);
5. выбор темы викторины (IP-основы и IP-адресация);
6. нажатие на кнопку "Играть";
7. запуск бесконечного цикла игр, условием выхода из которого является достижение небходимого количества игр;
8. ожидание поиска соперника;
9. обработка пяти вопросов раунда;
10. нажатие на кнопку "Играть снова";
11. после определенного количества игр закрыть браузер;

# Результаты работы


```
{
  "Question": "Маска подсети 255.255.192.0, выраженная в двоичном формате",
  "TrueFlag": 1,
  "TrueAnswer": "11111111 11111111 11000000 00000000",
  "IndexesIncorect": [],
  "Answers": []
},
{
  "Question": "Для 192.168.9.2/255.255.255.0, какая из следующих частей IP-адреса принадлежит хосту?",
  "TrueFlag": 0,
  "TrueAnswer": "",
  "IndexesIncorect": [
    1,
    3
  ],
  "Answers": [
    "2",
    "192.168",
    "192",
    "192.168.9"
  ]
},
```


Рис. 1: Фрагмент заполненной базы данных по теме IP-основы и IP-адресация

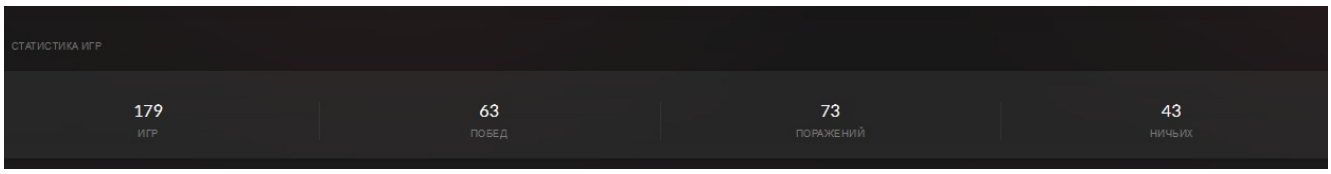


Рис. 2: Общая статистика пройденных игр (с учетом участия до использования бота)

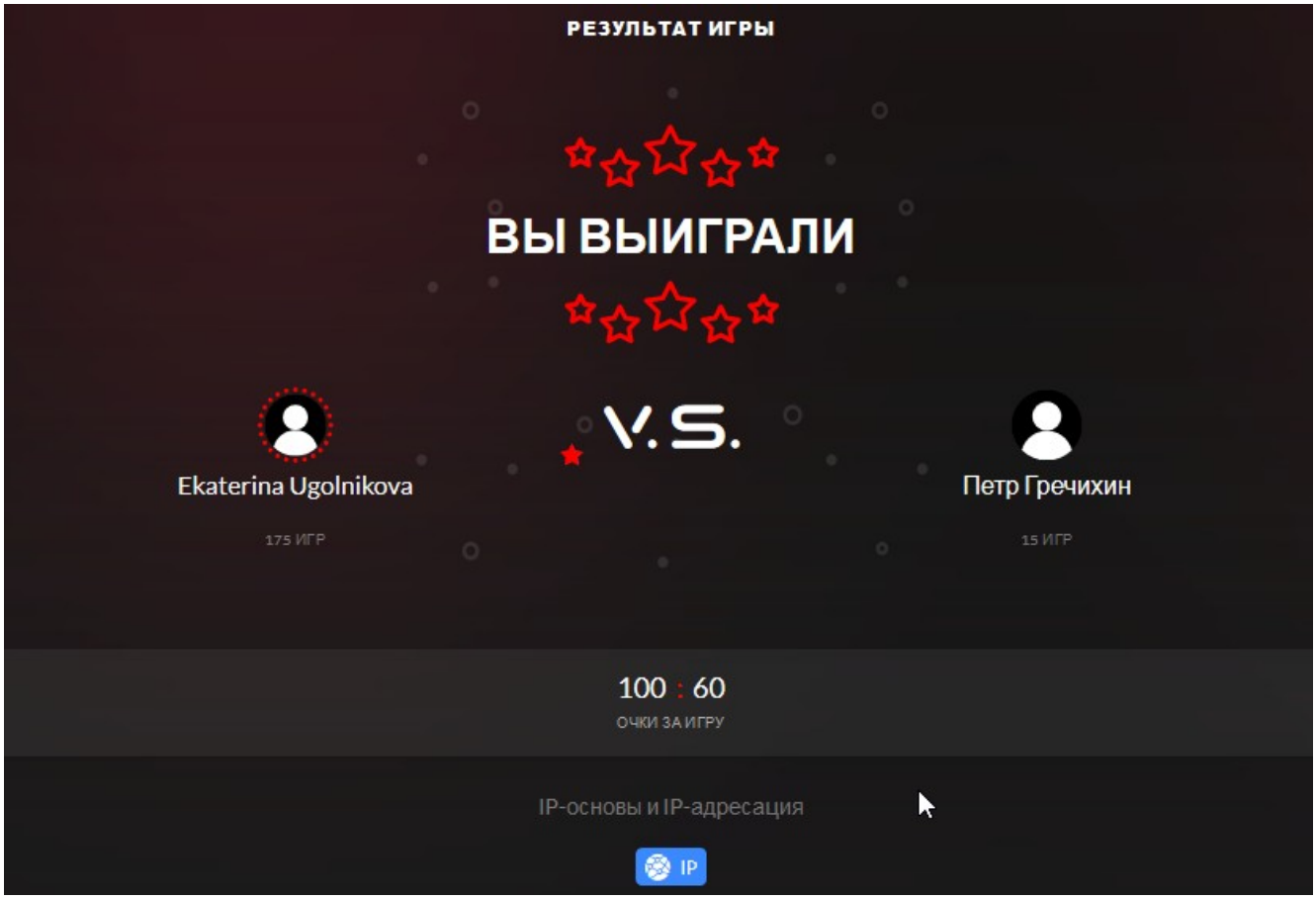


Рис. 3: Демонстрация результата работы бота в викторине

# Вывод

В результате лабораторной работы написан, настроен и протестирован бот для участия в викторине HonorCup. Тестирование проводилось в разделе IP на тему IP-основы и IP-адресация.

База вопросов заполнилась более чем на 70% примерно через 30 игр (105 вопросов всего), а скорректировалась до 95% верных примерно за 90 игр (100/105 верных ответов в базе на момент окончания тестирования).

Общее количество баллов на момент окончания тестирования: 655.

# Приложение 1

1. https://github.com/GiekoOlis/HonorCup-GameBot.git.

# Приложение 2

Листинг 1: Python HonorCupGameBot.py

```
# -*- coding: utf-8 -*-

# pip install selenium
# download geckodriver (https://github.com/mozilla/geckodriver/releases)

from selenium import webdriver
import time
import random
import json


def database_adjustment(database):
    for q in range(len(database)):
        if database[q]['TrueFlag'] == 1:
            database[q]['IndexesIncorect'].clear()
            database[q]['Answers'].clear()
        elif len(database[q]['IndexesIncorect']) == 3:
            indexes = [0, 1, 2, 3]
            for k in range(3):
                indexes.remove(database[q]['IndexesIncorect'][k])
            ind = indexes[0]
            database[q]['TrueFlag'] = 1
            database[q]['TrueAnswer'] = database[q]['Answers'][ind]
            database[q]['IndexesIncorect'].clear()
            database[q]['Answers'].clear()

    """
    QUESTION - Current Question[k] from DataBase
    CurrentAnswers - Answers from Current Question in GameRound
    HelpFunction for QuestionCoincide(*, *)
    """


def HelpFunction1(QUESTION, current_answers, i):
    for j in range(0, 4):
        if QUESTION['Answers'][i] == current_answers[j].text:
```

```
            current_answers[j].click()
            try:
                if browser.find_element_by_css_selector("div[
                    class='game__answer selected right']"):
                    QUESTION['TrueFlag'] = 1
                    QUESTION['TrueAnswer'] = current_answers[j
                        ].text
                    QUESTION['Answers'].clear()
                    QUESTION['IndexesIncorect'].clear()
                    break
            except:
                QUESTION['IndexesIncorect'].append(i)
    """
    QUESTION - Current Question[k] from DataBase
    CurrentAnswers - Answers from Current Question in
        GameRound
    When the desired question is found in Database
    Function to process the selected answer and correct the
        question in the database
    """
    def QuestionCoincide(QUESTION, current_answers):
        if QUESTION['TrueFlag'] == 1:
            for j in range(4):
                if QUESTION['TrueAnswer'] == current_answers[j].
                    text:
                    current_answers[j].click()
        else:
            if len(QUESTION['IndexesIncorect']) == 0:
                i = random.randint(0, 3)
                HelpFunction1(QUESTION, current_answers, i)
            else:
                indexes = [0, 1, 2, 3]
                for k in range(len(QUESTION['IndexesIncorect'])
                    ):
                    indexes.remove(QUESTION['IndexesIncorect'][k
                        ])
                i = random.choice(indexes)
                HelpFunction1(QUESTION, current_answers, i)


    def ADD_Question(QUESTIONS, Question, current_answers):
        k = random.randint(0, 3)
        current_answers[k].click()
        IndIncorrect = []
```

```
74          try:
75              if browser.find_element_by_css_selector("div[class
                    ='game__answer␣selected␣right']"):
76                  truef = 1
77                  TrueAnswer = current_answers[k].text
78                  IndIncorrect.clear()
79          except:
80              truef = 0
81              TrueAnswer = ""
82              IndIncorrect.append(k)
83          Quiz = {
84              "Question": Question,
85              "TrueFlag": truef,
86              "TrueAnswer": TrueAnswer,
87              "IndexesIncorect": IndIncorrect,
88              "Answers": [current_answers[0].text,
                    current_answers[1].text, current_answers[2].text,
                     current_answers[3].text]
89          }
90          QUESTIONS.append(Quiz)
91      """
92      QUESTIONS - Question DataBase from file
93      Question - Question from Current GameRound
94      CurrentAnswers - Answers from Current Question in
            GameRound
95      Function to find a question in the QUESTION Base
96      If the Question is missing from the database, it is added
             to the database
97      """
98      def QualitativeQuestion(QUESTIONS, Question,
            current_answers):
99          flag = 0
100         if len(QUESTIONS) != 0:
101             for i in range(len(QUESTIONS)):
102                 if QUESTIONS[i]['Question'] == Question:
103                     flag = 1
104                     QuestionCoincide(QUESTIONS[i],
                            current_answers)
105                     break
106                 else:
107                     continue
108         else:
109             ADD_Question(QUESTIONS, Question, current_answers)
110         if flag == 0:
```

```
111         ADD_Question(QUESTIONS, Question, current_answers)
112
113
114     def waiting():
115         while 1:
116             if browser.find_elements_by_class_name('
                game__answer'):
117                 break
118
119
120     IP_BaA = "IP_BasicsAndAddressing.json"
121
122
123     def save_questions():
124         with open(IP_BaA, "w", encoding="utf-8") as fi:
125             json.dump(QUESTIONS_database, fi, indent=2,
                ensure_ascii=False)
126
127
128     try:
129         QUESTIONS_database = json.load(open(IP_BaA, encoding='
            utf-8'))
130     except:
131         QUESTIONS_database = []
132
133     database_adjustment(QUESTIONS_database)
134     save_questions()
135     # get a token
136     TOKEN = 'https://quiz.honorcup.ru/app/?id=45815&sign=45
        bf30ca51861aff2af95aa6ecb42e5b'
137
138     # open a browser (Firefox)
139     browser = webdriver.Firefox(executable_path='geckodriver.
        exe')
140     browser.get(TOKEN)
141     time.sleep(2)
142
143     ##click battle_button
144     battle_button = browser.find_element_by_class_name('
        about__buttons')
145     battle_button.click()
146     time.sleep(2)
147
148     ##choose a category and theme
```

```
149     category = browser.find_elements_by_class_name('
            slider__item')
150     category[1].click()
151     time.sleep(2)
152     theme = browser.find_elements_by_class_name('
            profile__theme')
153     theme[0].click()
154     time.sleep(2)
155
156     # button Play
157     # Theme 0:
158     # categories_play_button = browser.find_element_by_xpath
            ('/html/body/app/div[1]/nomination/div/div/div[2]/div
            [3]/div[0]/div/div/div[2]/div')#('button-group-2x')
159     # Theme 1:
160     categories_play_button = browser.find_element_by_xpath('/
            html/body/app/div[1]/nomination/div/div/div[2]/div[3]/
            div[1]/div/div/div[2]/div')#('button-group-2x')
161     # Theme 2:
162     # categories_play_button = browser.find_element_by_xpath
            ('/html/body/app/div[1]/nomination/div/div/div[2]/div
            [3]/div[2]/div/div/div[2]/div')#('button-group-2x')
163     # Theme 3:
164     # categories_play_button = browser.find_element_by_xpath
            ('/html/body/app/div[1]/nomination/div/div/div[2]/div
            [3]/div[3]/div/div/div[2]/div')#('button-group-2x')
165
166     categories_play_button.click()
167     countGame = 0
168     while 1:
169         waiting()
170         for i in range(5):
171             round_question = browser.find_element_by_class_name
                    ('game__question-text')
172             round_answers = browser.find_elements_by_class_name
                    ('game__answer')
173             QualitativeQuestion(QUESTIONS_database,
                    round_question.text, round_answers)
174             while browser.find_elements_by_class_name('
                    game__answer'):
175                 continue
176             time.sleep(5)
177         countGame += 1
178         save_questions()
```

```
179             if countGame % 5 == 0:
180                 database_adjustment(QUESTIONS_database)
181                 save_questions()
182             if countGame % 20 == 0:
183                 break
184             restart = browser.find_element_by_xpath('/html/body/
                  app/div[1]/result/div/div/div[9]/div[1]')
185             restart.click()
186 
187     save_questions()
188     browser.close()
```