Guidelines for implementation

laboratory work No. 5 on the course " Web technologies"

**Server-side scripts. Form and processing of get and post requests** .

**Objective of the work** : to gain experience in writing and using server scripts to process data entered by the user.

**Rules for assessing works**

Completing the basic task is worth a maximum of 6 points. To get 10 points for the work, you must complete all the points from the "Extended Task" section.

Points will be deducted from the maximum possible grade in the following situations:

1. **Deadline.** The report is posted in the personal account after the deadline - minus 1 point every 2 weeks from the date recorded in the personal account. The date of the defense does not affect the grade received;
2. **Defense.** "Gaps" in knowledge during the oral defense of the work: grade "good" - minus 1 point, grade "satisfactory" - minus 2 points;
3. **Report.** The report contains minor violations of requirements - minus 1 point, several report items are missing - minus 2 points or more;
4. **Work.** The completed task lacks minor elements - minus 1 point, several points of the task are missing - minus 2 points or more.

Protection of work is possible only if there is a report in the personal account.

**Basic task**

Based on the examples provided, prepare a web page where the user enters several values and sends them to the server. After sending to the server, the entered data is printed on the page. The variant determines the type of request sent to the server ( get or post ) and the appearance of the displayed data. In addition, the variant specifies an additional form element that must be independently mastered and included in the form.

The questionnaire questions must reflect the site topic chosen in the first lab. It is prohibited to save sample fields except for the name.

Mandatory separate fields of the questionnaire form: 1) last name, 2) first name, 3) patronymic, 4) list with choice of values, 5) set of checkboxes , 6) set of radio buttons , 7) e-mail.

CSS styling to the form and to the page with the server response.

The server script is written in python .

When using the get method , the report should include the contents of the address bar after receiving the response from the server.

The page must be embedded into the site prepared in previous work.

**Extended Task**

1.. In the server's response, add a line formed according to the following template: "A.I. Ivanova is a student, wears medium shoe size and loves the Python language." (see the example below, Fig. 2 and 3). That is, on the page prepared by the server (Fig. 3), in addition to the table, there should also be a phrase written in which there are initials, surname and other values obtained from the questionnaire. There is no need to select the case and gender in the words.

2.. add a javascript script to the form that checks the entered values. Minimum checks: the fact that the fields are filled in; the text is checked for the absence of any characters except letters; the email is checked for the presence of the @ symbol. You can add your own checks.

3. . write the questionnaire data to a text file in the server script with accumulation of the result. Do not store the names of the fields.

**Procedure**

1. python shell
2. From Listing 1, create a python file (extension . py ) and write it to the main directory of the python shell . From the python shell , start the server. The server runs continuously.
3. Prepare a web page with a form (extension . htm ) according to the example in Listing 2. The web page must be located in the same directory as the file with the server. It is called in the browser address bar as follows: <http://localhost/forma.htm>You can edit the page in Notepad, and refresh the browser window to check the work.
4. Prepare a file with a server script (extension . py ) according to the sample of Listings 3, 4 or 5 depending on the variant. The file should be written to the cgi-bin directory , encoding utf -8. It is called by the server after pressing the button on the form. This file can also be edited in Notepad.
5. After finishing work, stop the server.

**Contents of the report**

1. front page;
2. purpose of work;
3. assignment option: website topic and work option #5
4. when using the get method : the address bar after receiving a response from the server.
5. screenshots: original page and page generated by the web server.
6. listings of two files (web page with form and server script)
7. conclusions on the work.

**Working with Python**

The simplest Python editor shell can be downloaded from the site <https://www.python.org/downloads/>, choose version 3.x . When working in this shell, two windows are used: the interpreter window (Fig. 1) and the text editor (Fig. 2). In the interpreter window, in the prompt line (>>>), you can write Python operators. When the program is executed in this window, output is displayed on the screen.

The text editor window is launched by the menu command File / New file (Fig. 2). Copy the web server text from Listing 1 into this window. In Python, the function of opening and closing brackets in compound operators is performed by indents from the beginning of the line. Therefore, all code operators from Listing 1 must be located strictly from the beginning of the line. Save the file with the . py extension and run it from the text editor ( F 5 button ). Messages about the server's operation will appear in the interpreter window (Fig. 3). The server is running constantly.

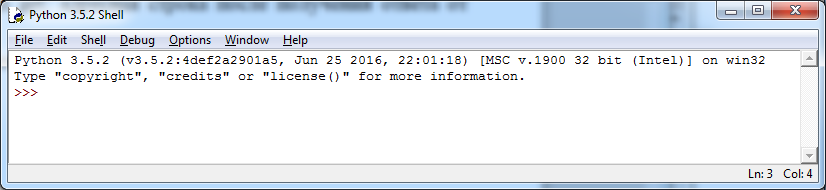


Fig. 1 Interpreter window for the Python language.

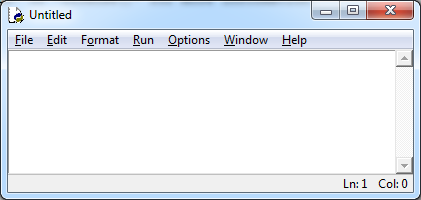


Fig. 2. Text editor window

**Listing 1. Web server in python . File serwerpy.py**

import os , sys

from http.server import HTTPServer , CGIHTTPRequestHandler

webdir = '.' # directory with HTML files and cgibin subdirectory for scripts

port = 80 # http://servername/ if 80, otherwise http://servername:xxxx/

if len ( sys.argv ) > 1: webdir = sys.argv [1] # command line arguments

if len ( sys.argv ) > 2: port = int( sys.argv [2]) # else By default ., 80

print( ' webdir "%s", port %s' % ( webdir , port))

os.chdir ( webdir ) # go to the root web directory

srvraddr = ('', port ) # hostname, port number

srvrobj = HTTPServer ( srvraddr , CGIHTTPRequestHandler )

srvrobj.serve\_forever () # serve clients until completion

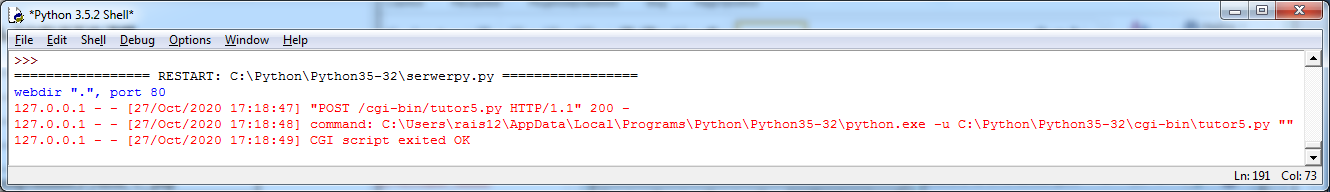


Fig. 3. Web server messages in the interpreter window.

**Comments on Listing 2**

**Form**

Forms are an element of a web page. They are used to send data to a web server. The form is enclosed in a paired tag <form> . Inside this tag are placed elements of the user interface - buttons, input fields, etc. Elements of the user interface can also be outside the form - in this case, in order for the interface to be useful, a script in the JavaScript language is attached to the elements .

Example first tag forms :

<FORM method="POST" action="/ cgi -bin/tutor5.py">

Basic form attributes: action, name, method.

action – contains the address ( url ) of the file that receives data from the form. In the example, this is a Python file tutor 5. py , located in the cgi - bin directory .

name – name of the form.

method – contains the name of the method ( get or post ) for transferring data to the web server. Data is transferred as a pair: element\_name \_ name = element\_value \_ value through the separator &.

In the description of the element "switch":

< input type = radio name = shoesize value =small>

The element's name is specified by the name attribute , and its value by the value attribute .

get method passes data from the form inside the url , this option means that the data being passed is open and accessible to everyone, as it is displayed in the browser's address bar. An example of passing data from a form when executing a get request :

[http://localhost/cgi-bin/horiztable.py?name=Аделина&shoesize=маленький&job=студент&language=Python&comment=люблю+программить](http://localhost/cgi-bin/horiztable.py?name=Аделина&shoesize=маленький&job=студент&language=Python&comment=люблю+программировать)

To get readable text when using Cyrillic, you need to go to one of the sites that encode/decode URLs with Cyrillic (for example, <https://www.design-sites.ru/utility/url-encoding.php>) and convert the text.

post method transfers data as a separate file. It is recommended to use it.

The form will submit data after a submit button is clicked . The value attribute here contains the text on the button. Example of a button:

< input type = submit value ="SEND">

There are different ways to create an interface element: using an unpaired <input> tag , in which the element type is specified via the type attribute , and using special tags. All interface elements have name and value attributes .

name attribute is the name of the interface element, required to access the element when sending data to the web server. The value attribute is the value of the element. For the text element , this is the default value, for the button, this is the inscription on it.

**Creating an interface using the <input> tag**

Main attributes: type , src .

**type** attribute specifies the type of the interface element. Value options:

button - button,

checkbox - checkbox,

file – a text field for entering a file name, there is a button to open a file selection dialog box.

hidden – an invisible element, but its data can be transferred to the server,

image – a graphic button of the " submit " type; the src attribute contains the name of the graphic file,

password – a text field for entering a password, similar to the text element ; all characters are hidden when printing.

radio - switch,

reset – button to set default values,

submit – a button that sends form data to the server,

text – text input field.

Examples of controls from Listing 3:

<input type= checkbox name =language value=Python>

<input type=radio name=shoesize value= small >

<input type=text name=name>

<input type=submit value=" SEND ">

**Other tags for interface elements**

The paired tag of the button <button> can contain other tags (text, image, table). To specify the type of button, use the type attribute with the values: button – a regular button, reset – cancellation of all changes in the form elements, submit – sending the form data, inside which the button is located, to the web server.

A drop-down list is created using a pair of <select> tags , inside which are placed <option> tags containing list items.

<select name=job>

<option> student <option> teacher <option> librarian

< option >other </ select >

Text input area <textarea> - a field for entering multiline text, may contain scroll bars. The cols and rows attributes define the size of the field in the number of characters horizontally and vertically.

< textarea name=comment cols=30 rows=2>

An invisible table is used to neatly place interface elements.

**Listing 2. The forma.htm file with the form.**

Launch file: <http://localhost/forma.htm>when web server is running

<HTML><TITLE>==Python==</TITLE>

<BODY>

<H1>User profile</H1>

<HR>

<FORM method="POST" action="/ cgi -bin/tutor5.py">

<H 3>Fill out the form and click the button</H3>

<P><TABLE>

<TR>

<TH align=right> Name :

<TD> <input type=text name=name>

<TR>

<TH align=right>Размер обуви:

<TD><table>

<td><input type=radio name=shoesize value=маленький>Маленький

<td><input type=radio name=shoesize value=средний>Средний

<td><input type=radio name=shoesize value=большой>Большой

</table>

<TR>

<TH align=right> Position :

<TD><select name=job>

<option> student

<option> teacher

<option> librarian

<option> other

</select>

<TR>

<TH align=right> Favorite language :

<TD><table>

<td><input type=checkbox name=language value=Python>Python

<td><input type=checkbox name=language value=C>C

<td><input type=checkbox name=language value=php>php

</table>

<TR>

<TH align=right>Комментарий:

<TD><textarea name=comment cols=30 rows=2>

введите текст</textarea>

<TR>

<TD colspan=2 align=center>

<input type=submit value=" SEND ">

</TABLE>

</FORM>

<HR>

</BODY></HTML>

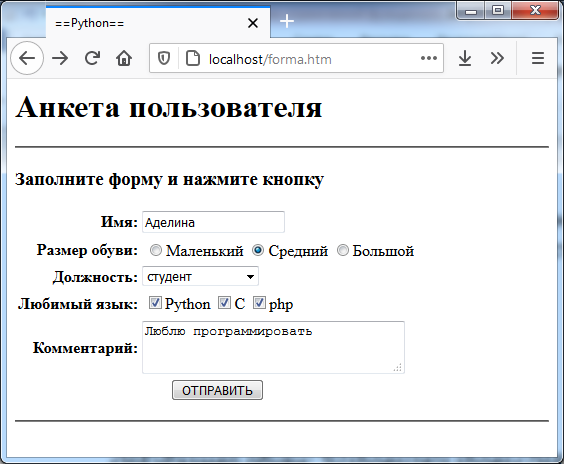


Fig. 2 Displaying the web page from Listing 2.

**Comments on Listing 3**

Python programming language

Python is similar to the C language in many ways, but has important differences:

1. declaration of variables is not required, the type is determined by the value assigned to the variable
2. the character " ;" terminates the statement, but the end of the line can also perform its role
3. There are no operator bracket symbols, they are replaced by an indent from the beginning of the page; the alignment of the beginning of each line is of fundamental importance.

A one-line comment is indicated by the # symbol.

import cgi , sys Connecting cgi and sys libraries

form = cgi . FieldStorage () The form variable is assigned data that is transferred from the form. A library function is used for this. The form variable is a dictionary, i.e. each element of the stored set of values consists of two parts - a key and a value. The key is the name of the interface element (the value of the name attribute ).

This code creates a web page. It does this gradually, using the Python function for output to the screen print (). The data that is output to the screen is organized in a table. The variable html1 is a string and contains the beginning of the web page. To write long text that is located on several lines, triple quotes are used.

After the first application of the print function (the line print ( " Content ­ type : text / html ")) a mandatory empty line is required here.

To display names in Russian, we form a list of names ll :

ll = [' name','size shoes','job title ', 'Favorite programming language','comment ']

A Python list is similar to an array in C in many ways, but there are differences: the list changes size during program execution, list elements can have different data types, numbering from the end of the list is possible (the last element has index -1), the list is a standard class and has a set of methods. The list sign is square brackets [].

The second list, data , will store the values retrieved from the dictionary variable form . We first form a list of several empty lines.

The variable i serves to index our lists.

Line for field in ('name', 'shoesize', 'job', 'language', 'comment'):

starts a loop through the 5 names of form interface elements. The for expression x in A means that the variable x will successively take on all values from the sequence A . This loop ends before the second word for .

Inside the loop, a check is performed to ensure that the form fields are filled in.

Line if not field in form:

checks that the current value of the field variable is not among the keys of the form dictionary .

If the name of the interface element is present, then it is checked whether the value of this interface element is a list (does not consist of several elements):

if not isinstance (form[field], list):

Multiple elements can contain a language field .

If there is only one value, it is added to the data list :

data[i] = form[field].value

If there are multiple values, a list values is generated . This uses a construct called a “list generator” ([ x . value for x in form [ field ]]). A list comprehension is a shorthand for writing a list loop. Next, the list elements are joined into a single string separated by commas. The string method . join () is used here. The resulting string is assigned to the data list .

values = [ x.value for x in form[field]]

data[ i ] = ', '.join (values)

After forming the data list , we output it to the web page through a loop:

for i in range( 5):

print ('<tr><td> %s </td> <td> %s </td></tr>'% ( ll [ i ], data[ i ]))

The range (5) function generates indices into lists, from 0 to 4. The % s signs in the output string indicate the positions where the value specified after the % separator (%( ll [ i ], data [ i ])) will be output. The s indicates that the value will be output as a string.

**Listing 3** . Output to a vertical table. File tutor 5.py

#output to table

import cgi , sys

form = cgi.FieldStorage () # retrieve data from form

print( " Contenttype : text/html ") # plus empty line

html1 = """

<TITLE>table with questionnaire</TITLE>

<H1>User profile</H1>

< table border =2> < tr > < td >Field Name</ td >< td >Value</ td > </ tr >

"""

# print table header

print (html1)

ll = [' name','size shoes','job title ', 'Favorite programming language','comment ']

data = ['','','','','']; i=0

for field in ('name', 'shoesize', 'job', 'language', 'comment'):

if not field in form:

data[i] = '(unknown)'

else:

if not isinstance(form[field], list):

data[i] = form[field].value

else:

values = [x.value for x in form[field]]

data[i] = ', '.join(values)

i +=1

for i in range( 5):

print ('<tr><td> %s </td> <td> %s </td></tr>'% ( ll [ i ], data[ i ]))

print ( ' </table> ')

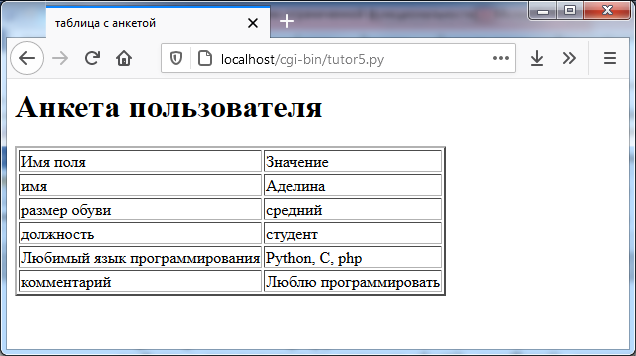


Fig. 3 Server response. Output of the entered questionnaire data in the form of a vertical table.

**Comments on Listing 4**

Unlike Listing 3, here the variable data is a dictionary in which the key is the name of the interface element and the value is the data entered by the user. The general logic of the algorithm is the same.

html line contains a web page template, into which the user data will be added to the places marked with the % sign in the last operator.

When working with a dictionary, an element is accessed by the key value:

data[field] = '(unknown)'

**Listing 4** . Output line by line . File tutor5.py

import cgi , sys

form = cgi . FieldStorage () # retrieve data from form

print( " Contenttype : text/html ") # plus empty line

html = """

<TITLE>tutor5.py</TITLE>

<H1>User Profile</H1> <HR>

<H4> You name : %(name)s</H4>

<H4> Size shoes : %(shoesize)s</H4>

<H4> Yours job : %(job)s</H4>

<H4> Beloved language : %(language)s</H4>

<H4> Comments :</H4>

<P>%(comment)s</P>

<HR>"""

data = {}

for field in ('name', 'shoesize', 'job', 'language', 'comment'):

if not field in form:

data[field] = '(unknown)'

else:

if not isinstance(form[field], list):

data[field] = form[field].value

else:

values = [x.value for x in form[field]]

data[field] = ' and '.join(values)

print(html % data)

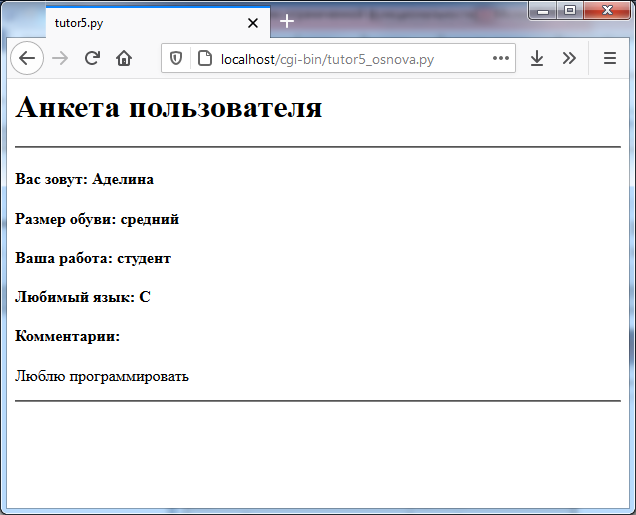


Fig. 4 Server response. Output of the entered questionnaire data line by line

**Comments on Listing 5**

The program elements included in the code in Listing 5 have already been explained in the previous comments.

**Listing 5** . Output to a horizontal table. File tutor 5.py

#output to table

import cgi , sys

form = cgi.FieldStorage () # retrieve data from form

print( " Contenttype : text/html ") # plus empty line

html1 = """

<TITLE>table with questionnaire</TITLE>

<H1>User profile</H1>

< table border =2> < tr >

"""

print (html1)

# print table header

ll = ['имя','размер обуви','работа', 'язык','комментарий']

for head in ll:

ss = '<td>'+head+'</td>'

print ( ss)

print ('</tr> <tr>')

data = ['','','','','']; i=0

for field in ('name', 'shoesize', 'job', 'language', 'comment'):

if not field in form:

data[i] = '(unknown)'

else:

if not isinstance(form[field], list):

data[i] = form[field].value

else:

values = [x.value for x in form[field]]

data[i] = ' and '.join(values)

i+=1

for el in data:

print ('<td> %s </td>'% el)

print ('</tr> </table>')

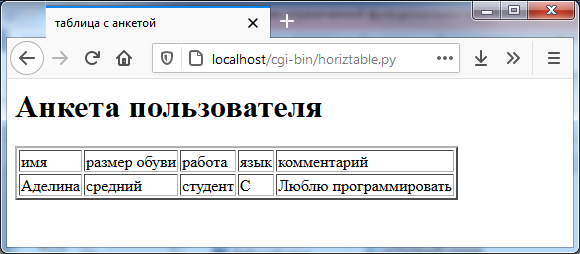


Fig. 5 Server response. Output of the entered questionnaire data in the form of a horizontal table.

Table 1. **Task options. Lr 5**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Method** | **Design of data output** | **Additional form element** |
| 1 | Get | Line by line | file |
| 2 | Get | Horizontal table | file |
| 3 | Get | Vertical table | file |
| 4 | Post | Line by line | file |
| 5 | Post | Horizontal table | file |
| 6 | Post | Vertical table | file |
| 7 | Get | Line by line | hidden |
| 8 | Get | Horizontal table | hidden |
| 9 | Get | Vertical table | hidden |
| 10 | Post | Line by line | hidden |
| 11 | Post | Horizontal table | hidden |
| 12 | Post | Vertical table | hidden |
| 13 | Get | Line by line | image |
| 14 | Get | Horizontal table | image |
| 15 | Get | Vertical table | image |
| 16 | Post | Line by line | image |
| 17 | Post | Horizontal table | image |
| 18 | Post | Vertical table | image |
| 19 | Get | Line by line | password |
| 20 | Get | Horizontal table | password |
| 21 | Get | Vertical table | password |
| 22 | Post | Line by line | password |
| 23 | Post | Horizontal table | password |
| 24 | Post | Vertical table | password |
| 25 | Get | Line by line | reset |
| 26 | Get | Horizontal table | reset |
| 27 | Get | Vertical table | reset |
| 28 | Post | Line by line | reset |
| 29 | Post | Horizontal table | reset |
| 30 | Post | Vertical table | reset |

Distribution of options for laboratory work No. 5 in the course " Web technologies"

Spring 2025

|  |  |  |
| --- | --- | --- |
| Option | Gr 4231 | Gr 4232 |
| 1 | Alekseev Leonid Alekseevich | Baranov Daniil Grigorievich |
| 2 | Alin Daniil Sergeevich | Kozlov Anton Vyacheslavovich |
| 3 | Beldenko Sofia Eniseevna | Konovalova Anastasia Vasilevna |
| 4 | Vasiliev Svyatoslav Sergeevich | Kostylev Vladimir Antonovich |
| 5 | Gavrilov Dmitry Vasilievich | Krikovtsov Yuriy Alekseevich |
| 6 | Ermakov Vyacheslav Andreevich | Kuznetsov Stepan Dmitrievich |
| 7 | Efremov Andrey Sergeevich | Kulagina Polina Andreevna |
| 8 | Ilyin Fyodor Aleksandrovich | Kurish Mikhail Viktorovich |
| 9 | Kashkolda Roman Sergeevich | Lavrov Svyatoslav Aleksandrovich |
| 10 | Kirillov Ruslan Yurevich | Lapin Yaroslav Anatolievich |
| 11 | Kozlov Artem Borisovich | Mayorov Arseniy Andreevich |
| 12 | Kozyrev Nikita Aleksandrovich | Andrey Vladimirovich Matveev |
| 13 | Kondratyev Kirill Vadimovich | Grigory Petrovich Matyushkov |
| 14 | Kochemirov Maksim Alekseevich | Makhmudova Milena Ilyasovna |
| 15 | Ovchinnikova Liliya Nikolaevna | Melnik Matvey Ilyich |
| 16 | Potapov Sergey Antonovich | Mikhailov Daniil Igorevich |
| 1 | Prourzin Pavel Sergeevich | Nefediev Ilya Innokentievich |
| 2 | Timofeev Aleksandr Vyacheslavovich | Ostrovsky Maksim Borisovich |
| 3 | Tkachev Ivan Aleksandrovich | Porokhnyak Mikhail Dmitrievich |
| 4 | Ushakov Daniil Pavlovich | Saveliev Vladislav Maksimovich |
| 5 | Cheverda Ilya Aleksandrovich | Salaev Nikita Leonidovich |
| 6 | Chernykh Daria Yurevna | Semenov Aleksandr Alekseevich |
| 7 | Chistyakova Kristina Aleksandrovna | Sotov Vadim Sergeevich |
| 8 | Shalya Vladislav Yurevich | Spitsov Andrey Nikolaevich |
| 9 | Shvedov Egor Aleksandrovich | Utkina Ekaterina Alekseevna |
| 10 | El- Heiba Denis Fadi Abdallaevich | Dmitry Andreevich Tsybin |
| 11 | Belchikov Nikita Andreevich | Yuzvenko Alina Dmitrievna |
| 12 | Saidov Sadi Tolibovic | Yuragin Nikita Valerevich |
| 13 | Mazur Bogdan Vladislavovich | Chemodanova Victoria Nikolaevna |

|  |  |  |
| --- | --- | --- |
| Option | Gr 4233 |  |
| 1 | Anufriev Maksim Antonovich |  |
| 2 | Bukaev Sergey Artemovich |  |
| 3 | Gozhev Dmitry Sergeevich |  |
| 4 | Golanova Sofia Vladimirovna |  |
| 5 | Grigoriev Danila Alekseevich |  |
| 6 | Dalibaeva Alina Jamalidinovna |  |
| 7 | Erygin Daniil Yurevich |  |
| 8 | Zorikhin Kirill Nikolaevich |  |
| 9 | Zubkov Ilya Yaroslavovich |  |
| 10 | Zuev Ruslan Dmitrievich |  |
| 11 | Kostash Renat Vadimovich |  |
| 12 | Kotkov Vladislav Vladimirovich |  |
| 13 | Kuznetsov Vladislav Dmitrievich |  |
| 14 | Larionovsky Roman Eduardovich |  |
| 15 | Anna Aleksandrovna Larchenkova |  |
| 16 | Luzanov Nikita Alekseevich |  |
| 1 | Magerov Anton Sergeevich |  |
| 2 | Osinkin Evgeniy Antonovich |  |
| 3 | Pavlov Sergey Romanovich |  |
| 4 | Patrukhin Alexey Ilyich |  |
| 5 | Romanovsky Kirill Arkadievich |  |
| 6 | Sadriev Amir Radimovich |  |
| 7 | Simonov Dmitry Alekseevich |  |
| 8 | Smirnov Vladislav Dmitrievich |  |
| 9 | Tipeeva Adelina Rinatovna |  |
| 10 | Chavychalov Maksim Evgenevich |  |
| 11 | Stadnik Evgeniy Petrovich |  |
| 12 | Erkinov Donior Bakhtior Ugli |  |

|  |  |  |
| --- | --- | --- |
| Option | Gr 4236 | Gr 4331 |
| 1 | Atyutskiy Nikita Mikhailovich | Basavin Igor Vladimirovich |
| 2 | Batin Vladislav Sergeevich | Berditsky Dmitry Igorevich |
| 3 | Borsch Nikita Sergeevich | Vartanyan David Eduardovich |
| 4 | Zhukovsky Sergey Alexandrovich | Goldashevsky Nikita Stepanovich |
| 5 | Zavgorodniy Alexander Romanovich | Denisov Erofey Pavlovich |
| 6 |  | Zinoviev Daniil Artemovich |
| 7 | Karpova Maria Evgenevna | Karlov Vsevolod Grigorievich |
| 8 | Kelbina Ksenia Anatolievna | Kostin Maksim Yurevich |
| 9 | Kolesnikova Larisa Eduardovna | Kumpel Artyom Gennadievich |
| 10 | Kudryashov Roman Sergeevich | Kushigina Anastasia Antonovna |
| 11 | Litvinova Anastasia Mikhailovna | Lambrecht Mikhail Olegovich |
| 12 | Lukina Anastasia Andreevna | Lekomtseva Yulia Maksimovna |
| 13 | Lulakov Daniil Feliksovich | Malenkov Mikhail Sergeevich |
| 14 | Machulsky Daniil Romanovich | Mikadze Victoria Valerevna |
| 15 | Nikolaev Artyom Aleksandrovich | Novitsky Roman Aleksandrovich |
| 16 | Panin Arthur Konstantinovich | Orlov Bogdan Ivanovich |
| 1 | Polpudnikov Vladislav Olegovich | Presnyakov Egor Stepanovich |
| 2 | Suprunenko Georgiy Aleksandrovich | Sergeeva Elena Aleksandrovna |
| 3 | Schukin Anton Aleksandrovich | Smirnov Konstantin Alekseevich |
| 4 | Yuriev Roman Andreevich | Trubnikov Aleksandr Vyacheslavovich |
| 5 | Mwale Leeson | Khaziev Renat Ramazanovich |
| 6 | Panchenko Andrey Viktorovich | Chernov Dmitry Andreevich |
| 7 | Valieva Darina Rafailovna | Isupov Dmitry Ruslanovich |
| 8 | Nazarova Daria Vadimovna |  |
| 9 |  |  |
| 10 |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Option | Gr 4332 | Gr 4333 |
| 1 | Artikulenko Vladimir Alekseevich | Argunov Artem Nikitich |
| 2 | Besedin Roman Mikhailovich | Bykov Aleksandr Sergeevich |
| 3 | Boldakova Alexandra Evgenevna | Vlasyuk Anastasia Nikolaevna |
| 4 | Budkovoy Ilya Pavlovich | Volodin Gleb Vladimirovich |
| 5 | Artem Olegovich Vasiliev | Volokitin Evgeniy Yurevich |
| 6 | Vereshchagin Nikita Borisovich | Voroshilov Daniil Nikolaevich |
| 7 | Tar Roman Ivanovich | Garshina Angelina Elizbarovna |
| 8 | Dobrynin Sergey Dmitrievich | Gerasimov Sergey Andreevich |
| 9 | Vladislav Katashov - | Dorofeev Aleksandr Vyacheslavovich |
| 10 | Kononenko Svyatoslav Vadimovich | Enichev Egor Olegovich |
| 11 | Korzhavin Konstantin Sergeevich | Kichkina Evgenia Denisovna |
| 12 | Krapivka Sergey Igorevich | Kolomeitsev Sergey Alekseevich |
| 13 | Lyutov Artem Alexandrovich | Korda Alexander Evgenievich |
| 14 | Martynov Alexander Alexandrovich | Lobanov Artem Olegovich |
| 15 | Migunov Maxim Sergeevich | Migunova Natalia Denisovna |
| 16 | Nikulina Yulia Eduardovna | Mogilatov Semyon Igorevich |
| 1 | Panaev Vladimir Vladimirovich | Dream Edward Sergeevich |
| 2 | Ponushkov Aleksandr Alekseevich | Suleimanov Rajab Ruslanovich |
| 3 | Saitkulov Dmitry Igorevich | Syrov Daniel Andzheevich |
| 4 | Toporina Anna Vladimirovna | Khatkevich Aleksandr Evgenevich |
| 5 | Chubar Dmitry Vasilievich |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
|  |  |  |
|  |  |  |