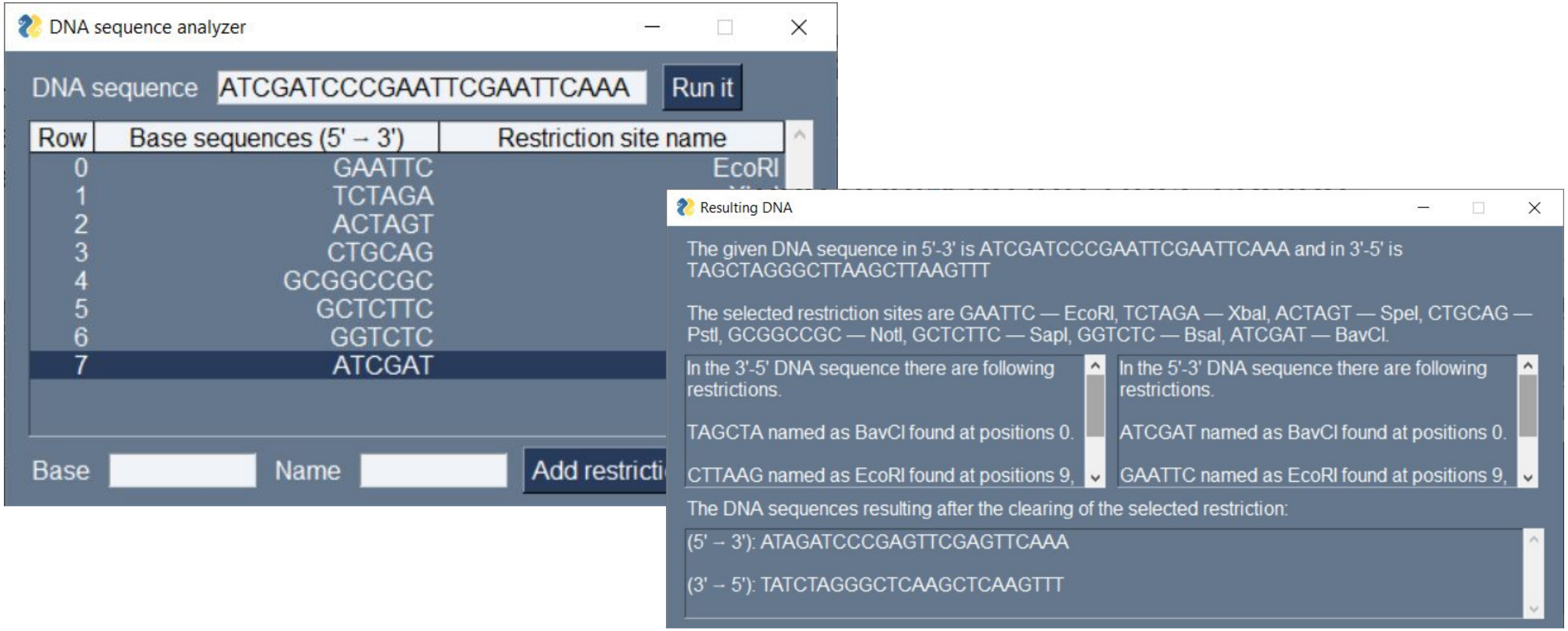
NAZARBAYEV UNIVERSITY

<CODE-ON>

Determination of restriction sites and their customized hiding within DNA sequence

Arman Bolatov, Polina Rakhalskaya, Daniil Filimonov, Darina Mukhamejanova
Team: NULL 404

Program's main menu and outputs



DNA sequence ATCGATCCCGAATTCGAATTCAAA

Row	Base sequences (5' → 3')	Restriction site name
0	GAATTC	EcoRI
1	TCTAGA	
2	ACTAGT	
3	CTGCAG	
4	GCGGCCGC	
5	GCTCTTC	
6	GGTCTC	
7	ATCGAT	

Base

Name

The given DNA sequence in 5'-3' is ATCGATCCCGAATTCGAATTCAAA and in 3'-5' is TAGCTAGGGCTTAAGCTTAAGTTT

The selected restriction sites are GAATTC — EcoRI, TCTAGA — XbaI, ACTAGT — SpeI, CTGCAG — PstI, GCGGCCGC — NotI, GCTCTTC — SapI, GGTCTC — BsaI, ATCGAT — BavCI.

In the 3'-5' DNA sequence there are following restrictions.

TAGCTA named as BavCI found at positions 0.

CTTAAG named as EcoRI found at positions 9,

In the 5'-3' DNA sequence there are following restrictions.

ATCGAT named as BavCI found at positions 0.

GAATTC named as EcoRI found at positions 9,

The DNA sequences resulting after the clearing of the selected restriction:

(5' → 3'): ATAGATCCCGAGTTCGAGTTCAAA

(3' → 5'): TATCTAGGGCTCAAGCTCAAGTTT

By NULL 404

What is special about our program?

The main feature of our program is that it not only determines the precise location of prohibited restriction sites according to iGEM standards, but also allows the user to hide these sites by altering the minimum number of nucleotides.

The algorithms are versatile and adaptable, so the researcher may add new restriction sites into the database and/or delete previous ones, so we are not limited with those provided by default.

The program will detect and change the DNA sequence according to the user’s preferences.

By NULL 404

What is special about our program?

An intuitive interface visually shows the restriction sites within the DNA sequence and annotates the precise name and position of the restriction site.

After determining the number and character of restriction site, the researcher can hide either all restriction sites or customize which restriction site they prefer to alter.

While the user enters DNA sequence for 5’-3’ direction, they receive the similar data output for the second strand with reversed direction.

The program makes sure that DNA alteration is similar to silent mutation and amino acids sequence encoded in coding frame remains the same.

By NULL 404

What is special about our program?

- Works for bacterial genome
- Inputs: the DNA sequence and restriction sites sequences to be further removed
- Can be applied for full gene sequence
- Assumes that DNA sequence input = coding region

By NULL 404

How-to run the program

- In the raw “DNA sequence”, enter the DNA sequence for analysis. The input sequence should be a coding region in 5’-3’ direction.

- By default, to determine the prohibited restriction sites and their location, no sites from an initial database are chosen. So you may choose all of them.

- To run the program, press the “Run” button.

- The resulting output identifies prohibited restriction sites within the given DNA sequence, and names their precise position.

- Also, the program presents similar data for the reversed (3’-5’) strand without additional input.

By NULL 404

How-to run the program

- By default, the program alters the nucleotide sequence and hides all found prohibited restriction sites within the given DNA sequence.

- If you want to customize which restriction sites you would like to remove, you may change the initial settings.

- To do so, close the program window to return to the main menu. Note: in the menu's table, the light blue sites will remain unchanged; the dark blue sites will be hidden.

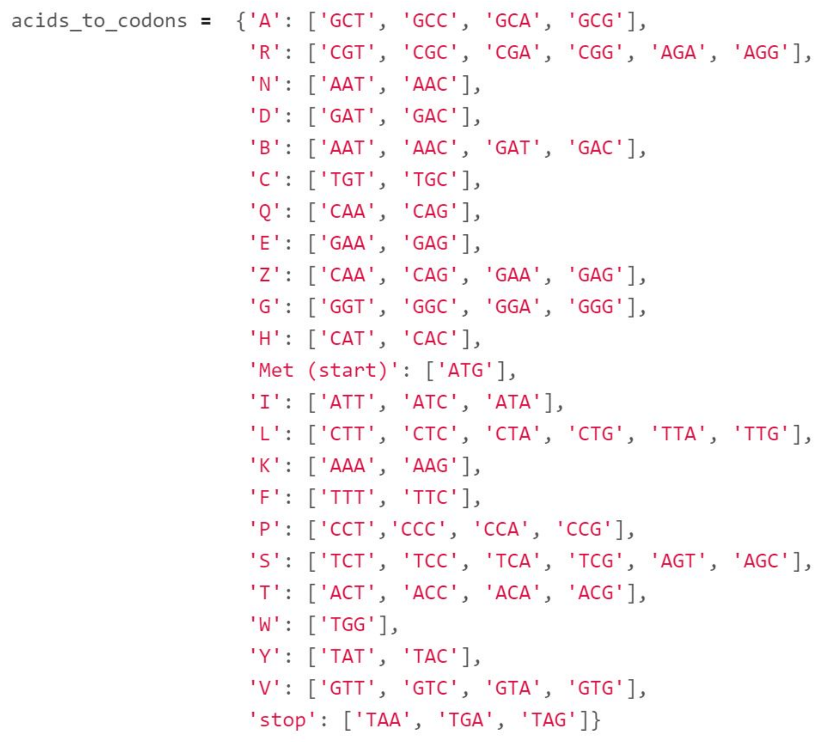
- Click on dark blue sites in the table of restriction sites to choose those to *keep* them within the DNA sequence.

- Press the "Run" button.

By NULL 404

Let’s deal with some algorithms!

For further convenience, we created supplementary dictionaries, that can convert amino acids’ name to codon name that this amino acid encodes.



```
acids_to_codons =  {'A': ['GCT', 'GCC', 'GCA', 'GCG'],
                     'R': ['CGT', 'CGC', 'CGA', 'CGG', 'AGA', 'AGG'],
                     'N': ['AAT', 'AAC'],
                     'D': ['GAT', 'GAC'],
                     'B': ['AAT', 'AAC', 'GAT', 'GAC'],
                     'C': ['TGT', 'TGC'],
                     'Q': ['CAA', 'CAG'],
                     'E': ['GAA', 'GAG'],
                     'Z': ['CAA', 'CAG', 'GAA', 'GAG'],
                     'G': ['GGT', 'GGC', 'GGA', 'GGG'],
                     'H': ['CAT', 'CAC'],
                     'Met (start)': ['ATG'],
                     'I': ['ATT', 'ATC', 'ATA'],
                     'L': ['CTT', 'CTC', 'CTA', 'CTG', 'TTA', 'TTG'],
                     'K': ['AAA', 'AAG'],
                     'F': ['TTT', 'TTC'],
                     'P': ['CCT','CCC', 'CCA', 'CCG'],
                     'S': ['TCT', 'TCC', 'TCA', 'TCG', 'AGT', 'AGC'],
                     'T': ['ACT', 'ACC', 'ACA', 'ACG'],
                     'W': ['TGG'],
                     'Y': ['TAT', 'TAC'],
                     'V': ['GTT', 'GTC', 'GTA', 'GTG'],
                     'stop': ['TAA', 'TGA', 'TAG']}
```

By NULL 404

Let’s deal with some algorithms!

Our first supplementary script is a “translator” of the DNA sequence to its complementary strand. That is the backbone of our program.

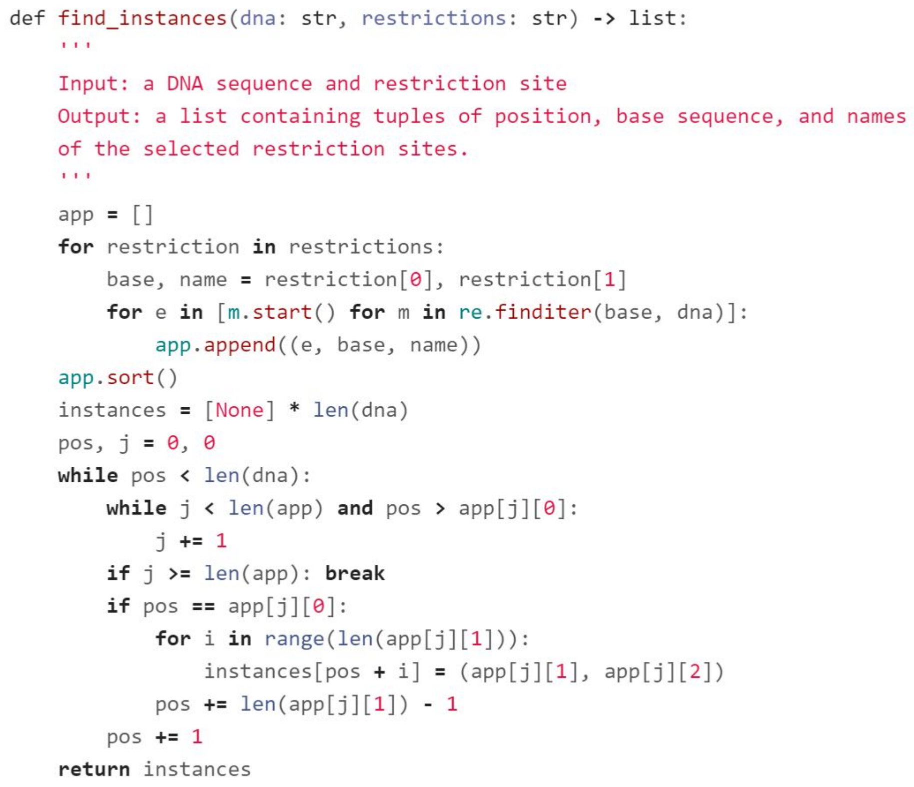
```python
def complement(dna: str) -> str:
    '''
    Input: a DNA sequences
    Output: its complementary sequences
    '''
    map = {"C": "G", "G": "C", "A": "T", "T": "A"}
    complement_dna = ""
    for nucleotid in dna:
        complement_dna += map[nucleotid]
    return complement_dna
```

By NULL 404

Let’s deal with some algorithms!

Our main script deals with the DNA sequences and restriction sites list. It starts with finding out restrictions in every single nucleotide bases strand, followed up by list “app” that includes the tuples of restriction code, name of restriction code, and its position in the sequence.

After that, the program goes through the given sequence again и creates a new array of an equal base length as the DNA itself. Our newly created array has the same length as the DNA. The numbered element demonstrates if the symbol with the same number belongs to the restriction or not. We will need this array a bit later!



```python
def find_instances(dna: str, restrictions: str) -> list:
    '''
    Input: a DNA sequence and restriction site
    Output: a list containing tuples of position, base sequence, and names
    of the selected restriction sites.
    '''
    app = []
    for restriction in restrictions:
        base, name = restriction[0], restriction[1]
        for e in [m.start() for m in re.finditer(base, dna)]:
            app.append((e, base, name))
    app.sort()
    instances = [None] * len(dna)
    pos, j = 0, 0
    while pos < len(dna):
        while j < len(app) and pos > app[j][0]:
            j += 1
        if j >= len(app): break
        if pos == app[j][0]:
            for i in range(len(app[j][1])):
                instances[pos + i] = (app[j][1], app[j][2])
            pos += len(app[j][1]) - 1
        pos += 1
    return instances
```

By NULL 404

Let’s deal with some algorithms!

The second supplementary script converts the DNA sequence to amino acid sequence. There we use the previously mentioned dictionary.

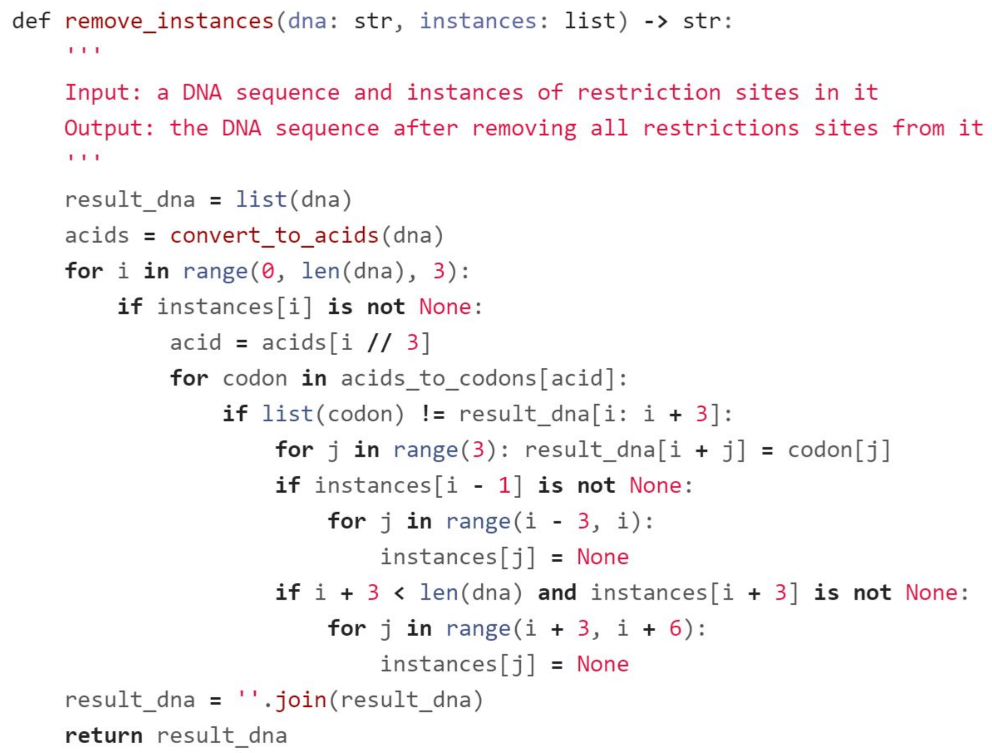
```python
def convert_to_acids(dna: str) -> list:
    '''
    Input: a DNA sequence
    Output: a sequence of aminoacids
    '''
    acids = []
    for i in range(0, len(dna), 3):
        if i >= len(dna) - 2: break
        codon = dna[i: i+3]
        acids.append(codons_to_acids[codon])
    return acids
```

By NULL 404

Let’s deal with some algorithms!

The next script hides the restriction sites after their detection in the given sequence.

The algorithm starts with the codons detection and if it finds out one with nucleotide from the restriction site list. Then the program looks for another codons that code the same nucleotide. After that the program changes the codon and deletes the restriction from the instances array. After such reorganization, we receive the DNA without restriction sites.

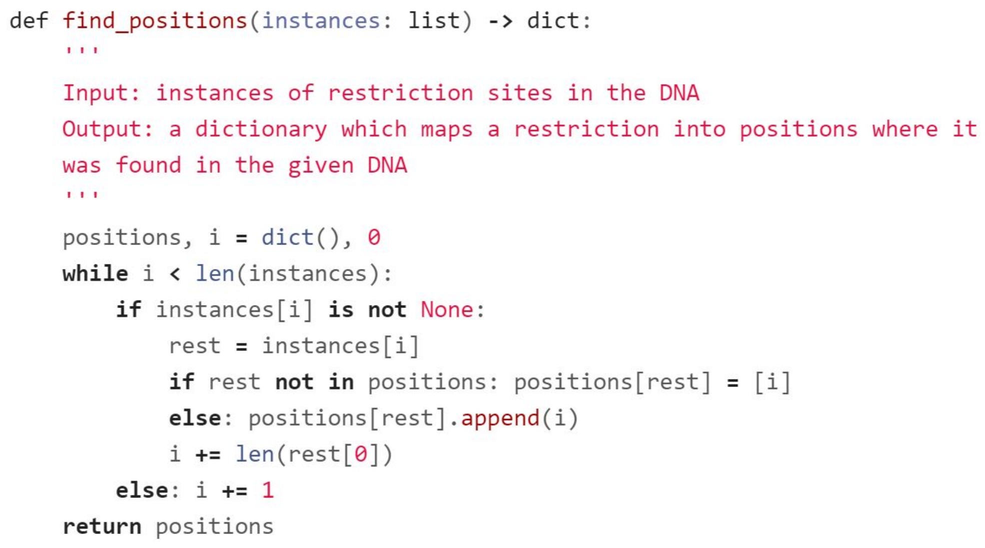


```python
def remove_instances(dna: str, instances: list) -> str:
    '''
    Input: a DNA sequence and instances of restriction sites in it
    Output: the DNA sequence after removing all restrictions sites from it
    '''
    result_dna = list(dna)
    acids = convert_to_acids(dna)
    for i in range(0, len(dna), 3):
        if instances[i] is not None:
            acid = acids[i // 3]
            for codon in acids_to_codons[acid]:
                if list(codon) != result_dna[i: i + 3]:
                    for j in range(3): result_dna[i + j] = codon[j]
                    if instances[i - 1] is not None:
                        for j in range(i - 3, i):
                            instances[j] = None
                    if i + 3 < len(dna) and instances[i + 3] is not None:
                        for j in range(i + 3, i + 6):
                            instances[j] = None
    result_dna = ''.join(result_dna)
    return result_dna
```

By NULL 404

Let’s deal with some algorithms!

Next script is needed to put the output data in suitable format: found restriction sites and their positions.



```python
def find_positions(instances: list) -> dict:
    '''
    Input: instances of restriction sites in the DNA
    Output: a dictionary which maps a restriction into positions where it
    was found in the given DNA
    '''
    positions, i = dict(), 0
    while i < len(instances):
        if instances[i] is not None:
            rest = instances[i]
            if rest not in positions: positions[rest] = [i]
            else: positions[rest].append(i)
            i += len(rest[0])
        else: i += 1
    return positions
```

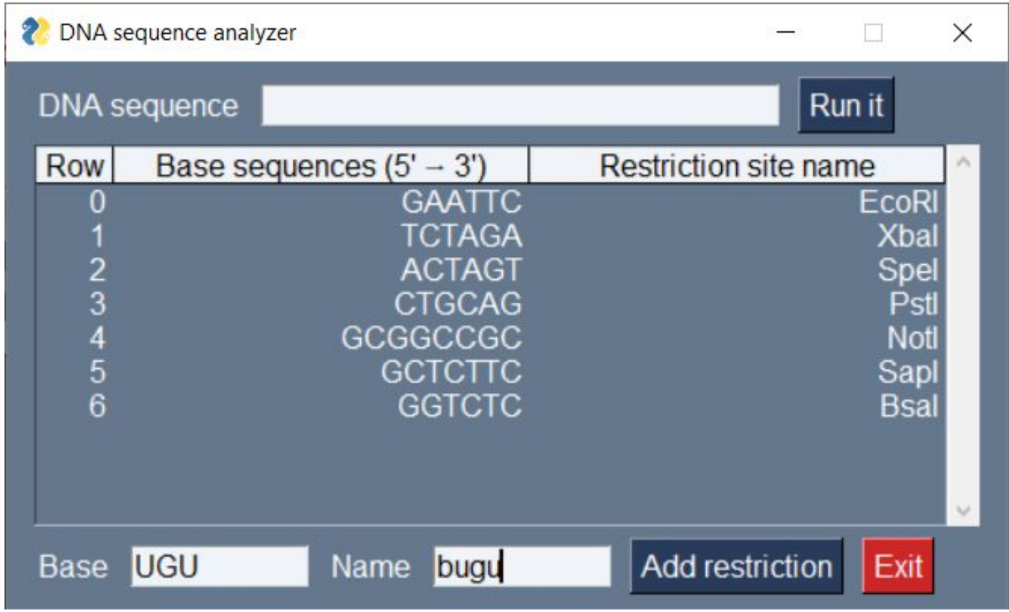
By NULL 404

… and finally an interface!

To make an interface for our project, we decided to use open source module PySimpleGUI.

There is a field “DNA sequence” in which we will input the sequence of our choice, table of restriction sites that user could choose on his own, “Base” and “Name” fields that made for adding new restriction site, and “Exit” button.

Let’s try to add a new restriction site in the table. For example, UGU and call it ‘bugu’.



DNA sequence analyzer

DNA sequence:

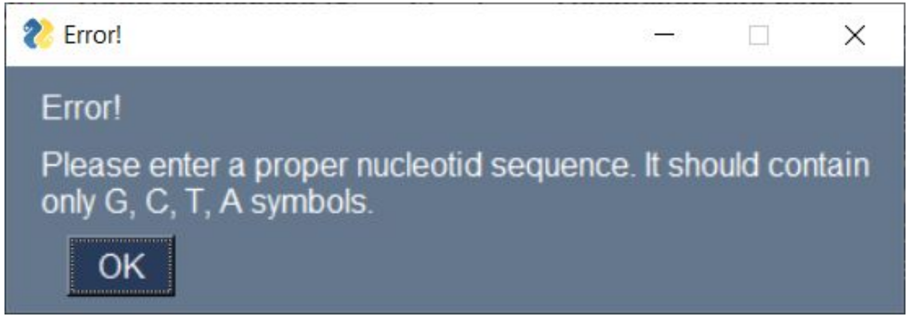
Row	Base sequences (5' → 3')	Restriction site name
0	GAATTC	EcoRI
1	TCTAGA	XbaI
2	ACTAGT	SpeI
3	CTGCAG	PstI
4	GCGGCCGC	NotI
5	GCTCTTC	SapI
6	GGTCTC	BsaI

Base: UGU Name: bugu

By NULL 404

… and finally an interface!

Oops… We get an error message: The letter U should not be in the code. Let's input the restriction site ATCGAT and call it BavCI.


Error!

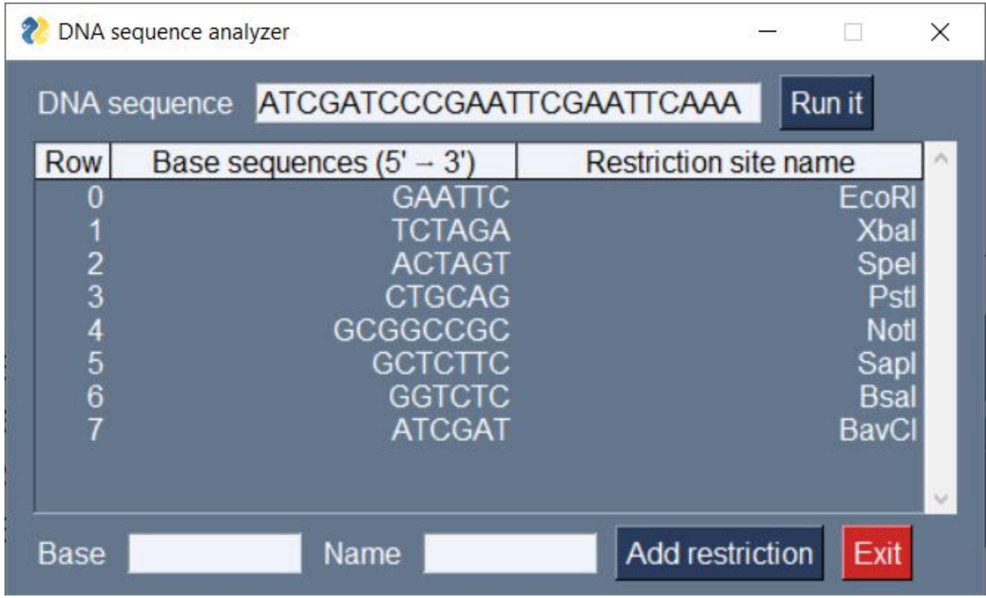
Error!

Please enter a proper nucleotid sequence. It should contain only G, C, T, A symbols.

By NULL 404

… and finally an interface!

As you can see, we successfully added a new restriction site. Now, let's test our program. Firstly, we need to choose restriction sites of our interest. We want to look at all restrictions that could be in the sequence, so we will choose everything.



DNA sequence ATCGATCCCGAATTCGAATTCAAA

Row	Base sequences (5' → 3')	Restriction site name
0	GAATTC	EcoRI
1	TCTAGA	XbaI
2	ACTAGT	SpeI
3	CTGCAG	PstI
4	GCGGCCGC	NotI
5	GCTCTTC	SapI
6	GGTCTC	BsaI
7	ATCGAT	BavCI

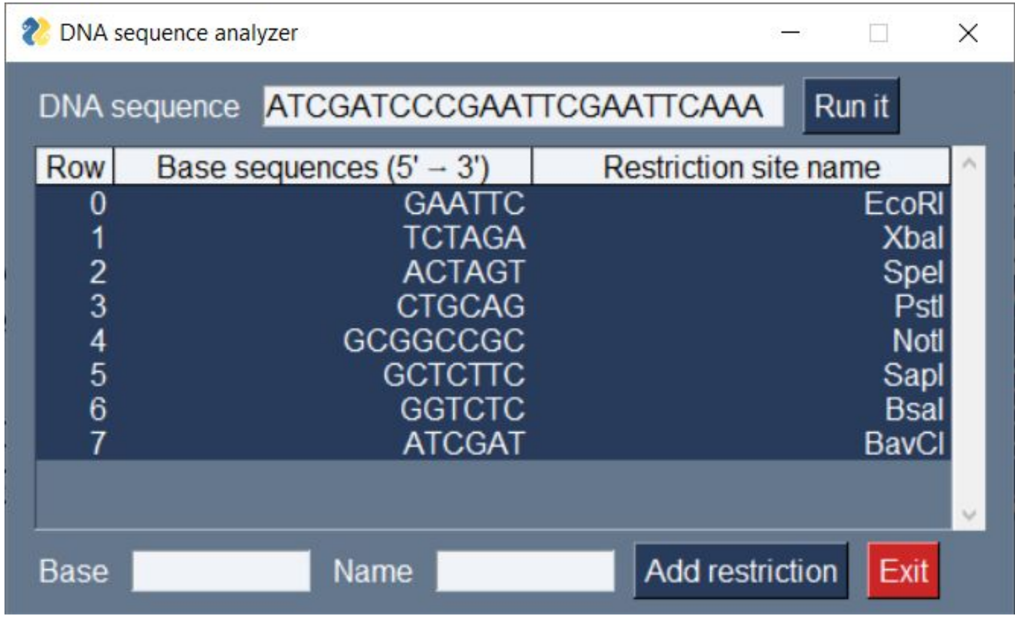
By NULL 404

… and finally an interface!

Now we need to input some sequence.

Let it be "ATC GAT CCC GAA TTC GAA TTC AAA"

And click the "Run" button!



DNA sequence: ATCGATCCCGAATTCGAATTCAAA

Row	Base sequences (5' → 3')	Restriction site name
0	GAATTC	EcoRI
1	TCTAGA	XbaI
2	ACTAGT	SpeI
3	CTGCAG	PstI
4	GCGGCCGC	NotI
5	GCTCTTC	SapI
6	GGTCTC	BsaI
7	ATCGAT	BavCI

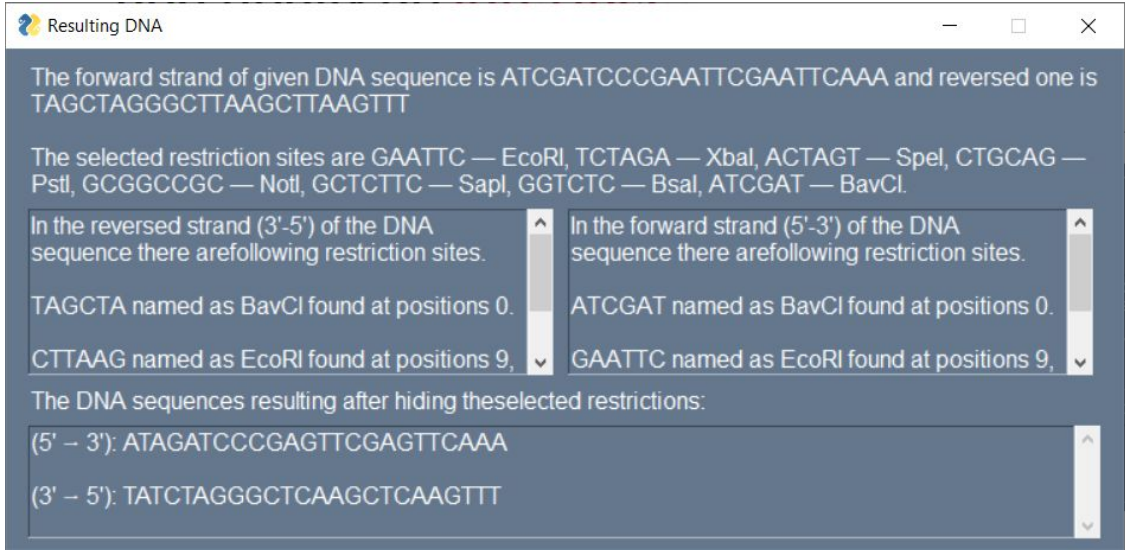
By NULL 404

… and finally an interface!

At first, we see the given sequence in 5’ →3’ and another complementary sequence. Below we can see the chosen restriction sites.

As it was turns out, there is ATCGAT/BavC on 0 position, and GAATTC /EcoRI on 9 and 15 positions. (Reading (5’ →3’))

After that, we could see the DNA sequence after the restriction sites were deleted by changing the base sites.



Resulting DNA

The forward strand of given DNA sequence is ATCGATCCCGAATTCGAATTCAAA and reversed one is TAGCTAGGGCTTAAGCTTAAGTTT

The selected restriction sites are GAATTC — EcoRI, TCTAGA — XbaI, ACTAGT — SpeI, CTGCAG — PstI, GCGGCCGC — NotI, GCTCTTC — SapI, GGTCTC — BsaI, ATCGAT — BavCI.

In the reversed strand (3'-5') of the DNA sequence there arefollowing restriction sites.

TAGCTA named as BavCI found at positions 0.

CTTAAG named as EcoRI found at positions 9,

In the forward strand (5'-3') of the DNA sequence there arefollowing restriction sites.

ATCGAT named as BavCI found at positions 0.

GAATTC named as EcoRI found at positions 9,

The DNA sequences resulting after hiding theselected restrictions:

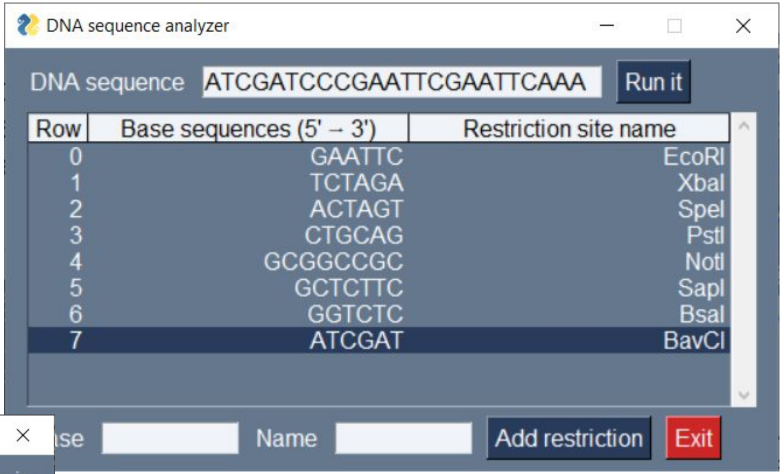
(5' → 3'): ATAGATCCCGAGTTCGAGTTCAAA

(3' → 5'): TATCTAGGGCTCAAGCTCAAGTTT

By NULL 404

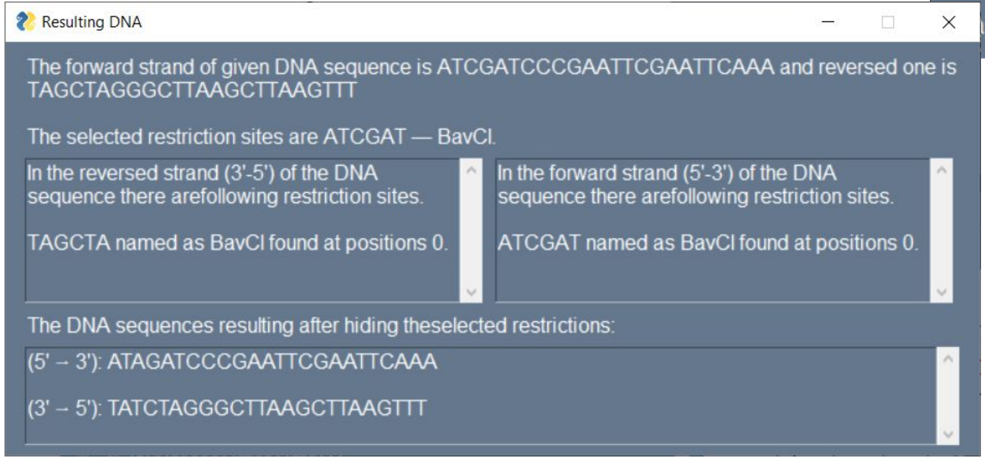
... and finally an interface!

What if we do not want to hide restriction site EcoRI, but only BavCI? That is simple! We go back, pick BavCI only, and и run the program again.



DNA sequence: ATCGATCCCGAATTCGAATTCAAA

Row	Base sequences (5' – 3')	Restriction site name
0	GAATTC	EcoRI
1	TCTAGA	XbaI
2	ACTAGT	SpeI
3	CTGCAG	PstI
4	GCGGCCGC	NotI
5	GCTCTTC	SapI
6	GGTCTC	BsaI
7	ATCGAT	BavCI



The forward strand of given DNA sequence is ATCGATCCCGAATTCGAATTCAAA and reversed one is TAGCTAGGGCTTAAGCTTAAGTTT

The selected restriction sites are ATCGAT — BavCI.

In the reversed strand (3'-5') of the DNA sequence there arefollowing restriction sites.

TAGCTA named as BavCI found at positions 0.

In the forward strand (5'-3') of the DNA sequence there arefollowing restriction sites.

ATCGAT named as BavCI found at positions 0.

The DNA sequences resulting after hiding theselected restrictions:

(5' – 3'): ATAGATCCCGAATTCGAATTCAAA

(3' – 5'): TATCTAGGGCTTAAGCTTAAGTTT

In our final answer, the third nucleotide was altered. We can prove using a table that both ATA и ATC code isoleucine, and that is why our result is correct.

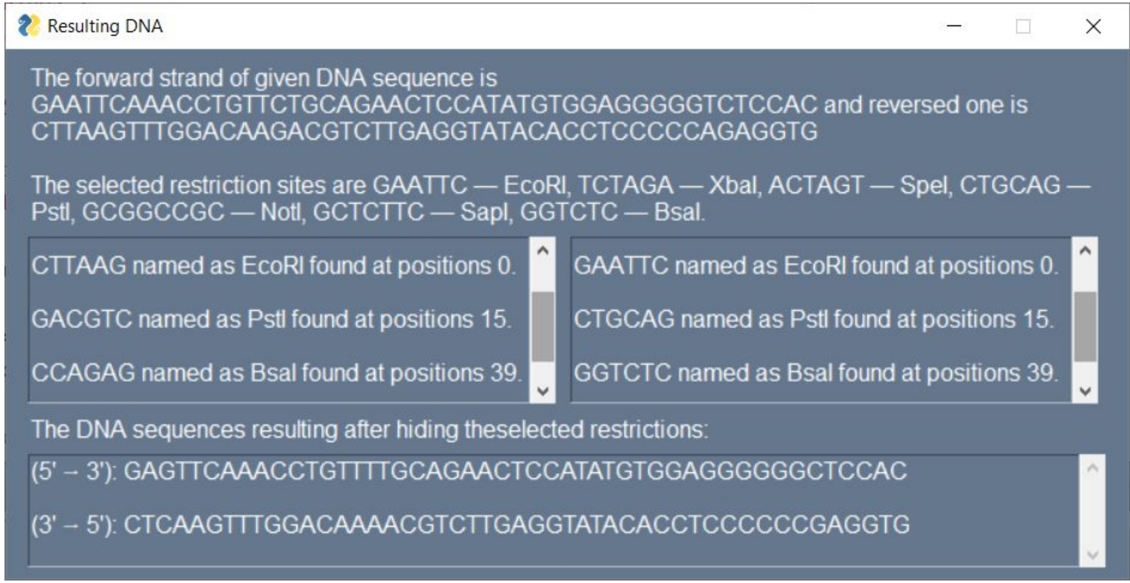
By NULL 404

Examples of inputs and outputs

The input DNA sequence contains an unknown number of restriction sites.

The programs detect three prohibited restriction sites: EcoRI, PstI, BsaI.

Then, by default, the program hides all restriction sites by altering nucleotide sequence within these sites.



```
The forward strand of given DNA sequence is
GAATTCAAACCTGTTCTGCAGAACTCCATATGTGGAGGGGTCTCCAC and reversed one is
CTTAAGTTTGGACAAGACGTCTTGAGGTATACACCTCCCCAGAGGTG

The selected restriction sites are GAATTC — EcoRI, TCTAGA — XbaI, ACTAGT — SpeI, CTGCAG —
PstI, GCGGCCGC — NotI, GCTCTTC — SapI, GGTCTC — BsaI.

CTTAAG named as EcoRI found at positions 0.      GAATTC named as EcoRI found at positions 0.

GACGTC named as PstI found at positions 15.      CTGCAG named as PstI found at positions 15.

CCAGAG named as BsaI found at positions 39.      GGTCTC named as BsaI found at positions 39.

The DNA sequences resulting after hiding theselected restrictions:

(5' – 3'): GAGTTCAAACCTGTTTTGCAGAACTCCATATGTGGAGGGGGCTCCAC

(3' – 5'): CTCAAGTTTGGACAAAACGTCTTGAGGTATACACCTCCCCCGAGGTG
```

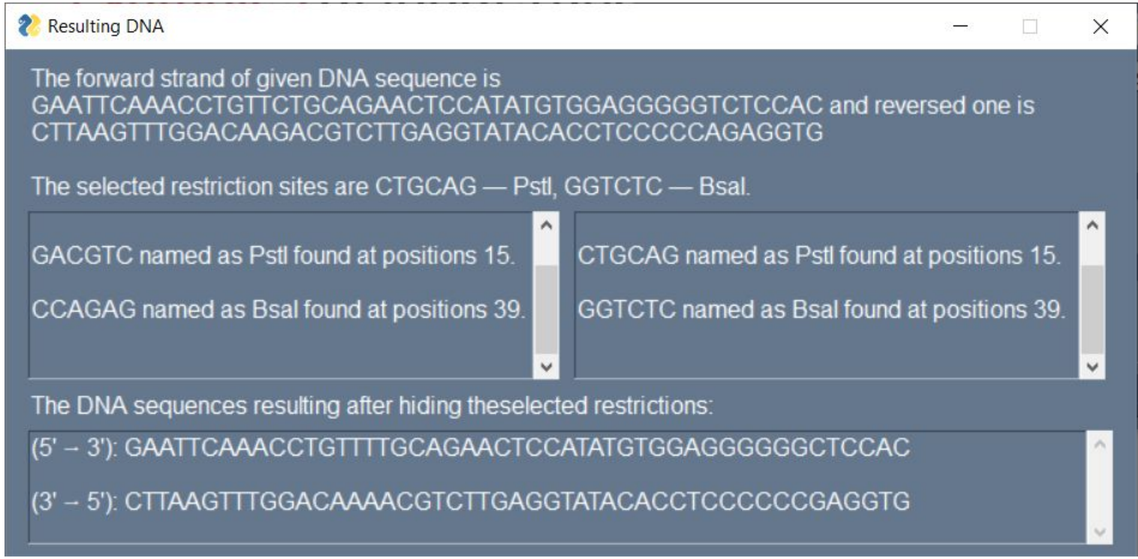
By NULL 404

Examples of inputs and outputs

The same DNA sequence contains three prohibited restrictions sites: EcoRI, PstI, BsaI (see the 20th slide).

This time, user prefers to hide only PstI, BsaI, but leave EcoRI unchanged. They select only PstI, BsaI from the main menu, willing to remove only those two. EcoRI is kept in the sequence (position 0).

The output contains the unchanged EcoRI site within the DNA sequence output.



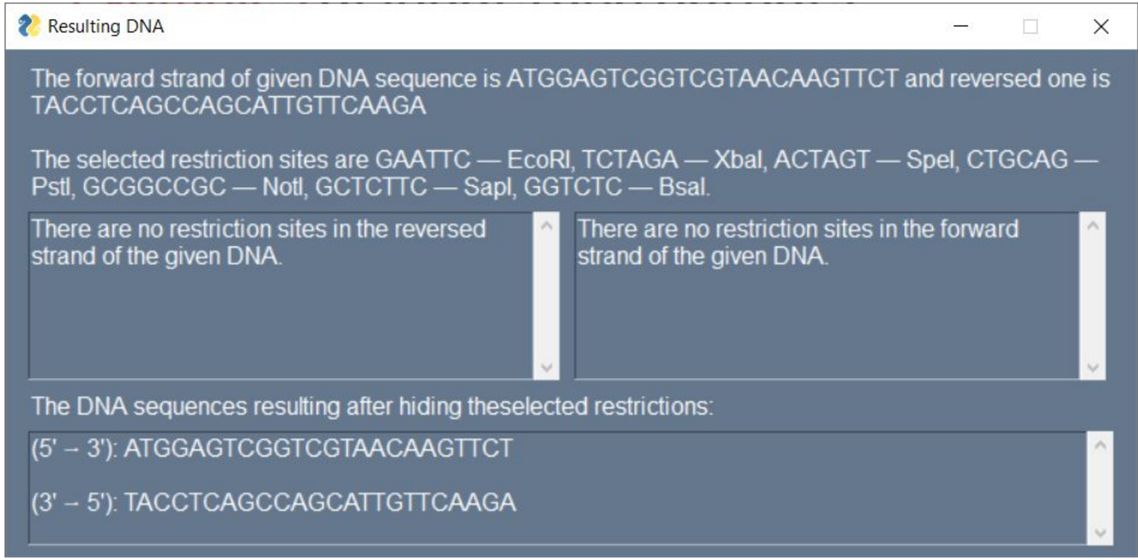
```
The forward strand of given DNA sequence is
GAATTCAAACCTGTTCTGCAGAACTCCATATGTGGAGGGGGTCTCCAC and reversed one is
CTTAAGTTTGGACAAGACGTCTTGAGGTATACACCTCCCCCAGAGGTG

The selected restriction sites are CTGCAG — PstI, GGTCTC — BsaI.

GACGTC named as PstI found at positions 15.      CTGCAG named as PstI found at positions 15.

CCAGAG named as BsaI found at positions 39.      GGTCTC named as BsaI found at positions 39.

The DNA sequences resulting after hiding theselected restrictions:
(5' → 3'): GAATTCAAACCTGTTTTGCAGAACTCCATATGTGGAGGGGGGCTCCAC

(3' → 5'): CTTAAGTTTGGACAAAACGTCTTGAGGTATACACCTCCCCCCGAGGTG
```

By NULL 404

Examples of inputs and outputs

The given sequence of DNA does not contain any prohibited restriction site.

The program neither detects any nor changes the DNA sequence.



The forward strand of given DNA sequence is ATGGAGTCGGTCGTAACAAGTTCT and reversed one is TACCTCAGCCAGCATTGTTCAAGA

The selected restriction sites are GAATTC — EcoRI, TCTAGA — XbaI, ACTAGT — SpeI, CTGCAG — PstI, GCGGCCGC — NotI, GCTCTTC — SapI, GGTCTC — BsaI.

There are no restriction sites in the reversed strand of the given DNA.

There are no restriction sites in the forward strand of the given DNA.

The DNA sequences resulting after hiding theselected restrictions:

```
(5' → 3'): ATGGAGTCGGTCGTAACAAGTTCT

(3' → 5'): TACCTCAGCCAGCATTGTTCAAGA
```

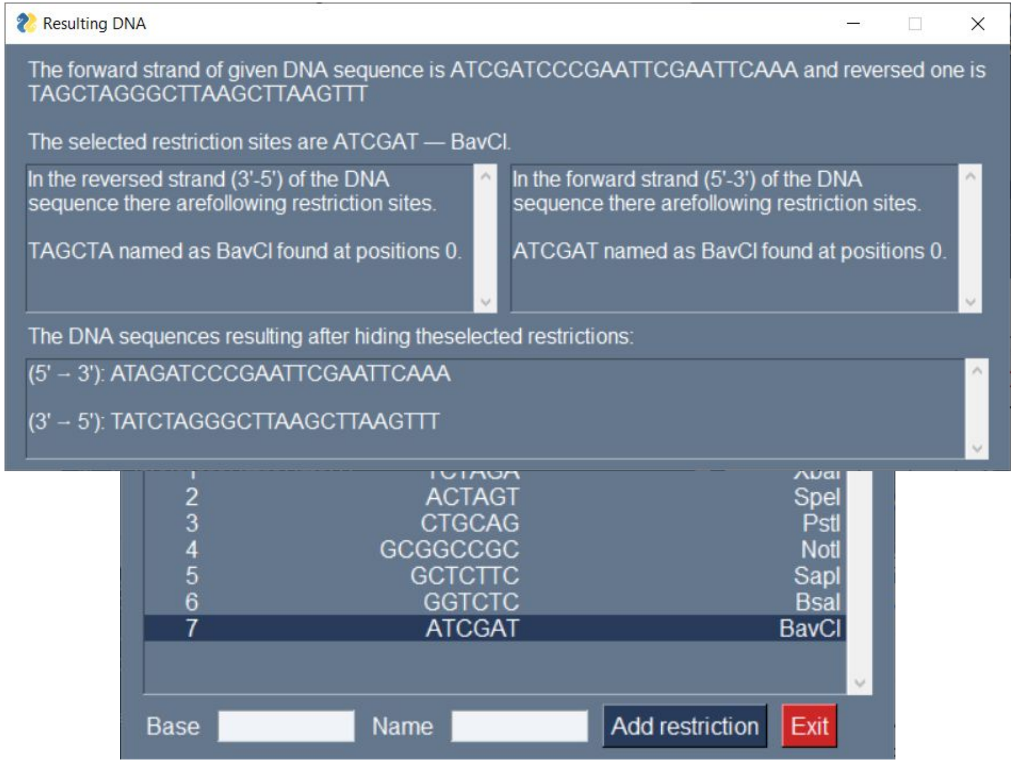
By NULL 404

Examples of inputs and outputs

The given sequence of DNA contains the restriction site not present in the main base of prohibited restriction sites.

The researcher may add any new restriction site (ATCGAT- BavCl) into the base, and the program will detect and alter it if that site is present within the sequence.

So its position is 0 in the DNA sequence and the output sequence has an altered site (ATAGAT), coding the same 2 amino acids (Ile,asp).



The forward strand of given DNA sequence is ATCGATCCCGAATTCGAATTCAAA and reversed one is TAGCTAGGGCTTAAGCTTAAGTTT

The selected restriction sites are ATCGAT — BavCl.

In the reversed strand (3'-5') of the DNA sequence there arefollowing restriction sites.

TAGCTA named as BavCl found at positions 0.

In the forward strand (5'-3') of the DNA sequence there arefollowing restriction sites.

ATCGAT named as BavCl found at positions 0.

The DNA sequences resulting after hiding theselected restrictions:

(5' → 3'): ATAGATCCCGAATTCGAATTCAAA

(3' → 5'): TATCTAGGGCTTAAGCTTAAGTTT

2	ACTAGT	Spel
3	CTGCAG	Pstl
4	GCGGCCGC	Notl
5	GCTCTTC	Sapl
6	GGTCTC	Bsal
7	ATCGAT	BavCl

By NULL 404

Possible modifications

For future development of the program, we consider:

- Insert additional database, so user may also create their account to save preferable restriction sites to be removed
- Make visualization of restriction maps and demonstrate restriction bars within the strand
- Adapt algorithms for more complex genes organization similar to eukaryotes

By NULL 404