Приложение 2. Демонстрация работы программы

Часть 1. Генератор ключей

In [1]:
```
# Сначала очистим директорию от артифактов предыдущих запусков кода

!rm -f encrypted.* decrypted.* *.public *.secret
!ls -l
```

```
total 1160
-rw-r--r-- 1 evgeny evgeny 432799 Apr 25 17:03 analyze.ipynb
-rw-r--r-- 1 evgeny evgeny  88091 Apr 25 17:05 demo.ipynb
-rw-r--r-- 1 evgeny evgeny 287142 Apr 23 17:18 demo.pdf
-rw-r--r-- 1 evgeny evgeny  35117 Apr 23 16:57 image.png
-rw-r--r-- 1 evgeny evgeny   4222 Apr 24 12:50 keygen.py
-rw-r--r-- 1 evgeny evgeny 306562 Apr 25 16:42 mayday.png
-rw-r--r-- 1 evgeny evgeny    408 Apr 20 15:36 open.txt
drwxr-xr-x 2 evgeny evgeny   4096 Apr 24 12:51 __pycache__
-rw-r--r-- 1 evgeny evgeny   5160 Apr 23 16:57 rsa.py
drwxr-xr-x 6 evgeny evgeny   4096 Apr 15 15:18 venv
```

In [2]:
```
# Запускаем генерацию ключевой пары
# Скрипт по-умолчанию создаёт пару ключей с названиями "key.pub"
# и "key.secret", или можно напрямую передать скрипту префикс и
# получить пару ключей с другими названиями

# Генератор ключевой пары принимает два парметра p_size и q_size
# Чтобы соблюсти условие значительной величины их разницы -
# задам им разную длину

from keygen import KeyGenerator

KeyGenerator(1024, 3072)

# Проверим создались ли файлы с ключами
!ls -l
```

```
total 1168
-rw-r--r-- 1 evgeny evgeny 432799 Apr 25 17:03 analyze.ipynb
-rw-r--r-- 1 evgeny evgeny  88091 Apr 25 17:05 demo.ipynb
-rw-r--r-- 1 evgeny evgeny 287142 Apr 23 17:18 demo.pdf
-rw-r--r-- 1 evgeny evgeny  35117 Apr 23 16:57 image.png
-rw-r--r-- 1 evgeny evgeny   4222 Apr 24 12:50 keygen.py
-rw-r--r-- 1 evgeny evgeny   1541 Apr 25 17:05 key.public
-rw-r--r-- 1 evgeny evgeny   2053 Apr 25 17:05 key.secret
-rw-r--r-- 1 evgeny evgeny 306562 Apr 25 16:42 mayday.png
-rw-r--r-- 1 evgeny evgeny    408 Apr 20 15:36 open.txt
drwxr-xr-x 2 evgeny evgeny   4096 Apr 24 12:51 __pycache__
-rw-r--r-- 1 evgeny evgeny   5160 Apr 23 16:57 rsa.py
drwxr-xr-x 6 evgeny evgeny   4096 Apr 15 15:18 venv
```

In [3]:
```
# Посмотрим содержимое key.pub
# Файл содержит строки в 16-м формате со значением
# e (открытый ключ) и n (модуль) разделенные знаком /

!cat key.pub
```

```
cat: key.pub: No such file or directory
```

In [4]:
```
# Посторим файл key.secret
# Файл содержит строки в 16-м формате со значением
# d (закрытый ключ) и n (модуль) разделенные знаком /

!cat key.secret
```

```
0X15971E2D9CA67D18B9B9D8F722F5744C04BDAA4FF8A5C22B363FB2CFEF8E2B7A1
9F5BD744B6726853FBFAAED45739CBE55AC25C3B2C37A8F9A99488604447389BF6A
B651BB26CAE65B374921062994385A0FC7F05FA6EFD6F5CB898E28F0D12E5C93C2C
C5874585568E8618F1C9EAB658A2F86DDE7A12E13AD82A55CA39B1D404D9C013915
1AD46A313F5A4BB996D91B22F50234CFCD0E9FCAC6F9FFA38F39040C8974ABD70D7
44099D96DDB1E61AC4DF80CC7BD8ADAF774D132707C5CBC38C1AC340F5521A88B21
D0E330399B398DF7BBDBD70D6BFF036CD6B5013AA4148CEDA3523FFE9EF71117910
A502D8B85582AB24FDCFAEEA22AC8899F1DE90C5D346E66AE71655F71945CFC6FC1
53A668A5B8C60E58B33A0EB86B42F8548130BC94CA82B28A0B1B11F187A47C2DB73
7F6E6B309BCA306894AE54BBD81F6196500A12FE06D855278325DB5F51498F651CC
558FA5AFB2615AD0847FDFA3AD30375AE0FB26E1208B6128ED625EF34D4520F61EE
4E42EF7D602FDB89CCF5E5E70B2A03A725EEAE43510B9E64B0747C9B669014A0BE6
6ED648F6AAA6ECCE91B5F63DB61CDE335D6D9F8F0762967C72EA820DBECBE01B8C3
F534F76EDD9FFE5F81846737FBEF4A78DEB6ED25A322588E3A7544C775B783E460A
4EC721FB81DDC439C67E6BCED7C73EDF1F41DBDF11535BA533EEC35B9B698B0E267
582FAA8054CABD28FFDF3/0X8AF90DE46B7ED7675DFB4FF39A315B3080E36A2ED28
11D596FCCEC3F23DAC69533EEF57225DEEFF492BBF70E63CA3FD2C02F926EC43137
F0E6459FE6DB31E0AF63B457FD4DD77366CE1249EA3479E7C5BD94A6CB1ED7CA22C
A170C5F5B60C4459A63176375CCBC0F251AAC3CA9DBE992E4CC9782D2A2F2990A34
CA6BD86BA9CBE25F68B1F0B1CBAAA0D1E0BEB0717728796A9BDD1EF1F97DF63E625
7FE7F57EE2420D3F8C62F031B57E6B7124423520FD831C7757E91E039E141014905
D6E74A5B88256ABE607998B14C2B5440F8199C0F15191F8C3367D935D42A3FF3024
350F41DE127C050D83E8A7FDFC9C322972E421091D19C1BAB418561FC319D2AD5E2
0A07B2868BD667B43E02210CD9C8FFB926B11DC7B3F725B023098B04A6301EDD79A
B93C1E21E92273908E7B8BDDC3424E145949C36ABD43886C5C493281B09C74C9999
8F570D343B0152B1DD2FF115E7FB86A5D8E78DD5DE36FE197F9BBEA13D9E2FF9A45
60A6FDEF2CC80B1372578FCCB35511E448E364980AB2633471D19128649D62B01B8
6CC37DE81E83C9F32D79FC69D5DB54995FD744CAA59000107C841DC2B74A2F912EA
70A8D9BE4E1DE2B858AF18003F2369A0870F83AFF2688E0F3970071AFA695D44CDD
04F0DF6AAC12EF06E86CD5DC91368BC736AEE25D05E0E1B6040B6C8EEFA4A38FFC4
F045FD352D76150E67F940205B0B67E1ADFED1202DF
```

In [5]:
```
# Создам объект класса RSA для шифрования файлов и
# передам ему ключ для шифрования (открытый)

from rsa import RSA

encryptor = RSA(public_key_path="key.public")
encryptor
```

Out[5]:
```
<rsa.RSA at 0x7f69c94ad890>
```

In [6]:
```
# Для наглядности вывожу в строчном представлении
# модели и её основные параметры. Видим что объект
# класса обладает только ключами для зашифрования

encryptor.__str__()
```

Out[6]:
{'public_key': '0X9BFA4C04A772DD0CD052F930680FB65DC1AB85954C4284DDF
9571AD8EBEBBEA3F82DC6B4CA1FDECA0A033941F40537E453550B41A3C241BD0D2A
8C07C9E518EFF90E6CCA816FACD93EDD8E178631270037BB6D4162472BF123FB0C1
10A8F40367B934D51EF85E94E0FD2B2AF71CF119A78633F95E6C64F832A150A5CE7
C323CC7E32B25279F7D5329FA6E01E7BF87AB17AE53E78B46E832CB9E547667EF74
783D2CEBD239C9B47B1AB6999BFFB9415F1959B018E0A6288003C1CF0945AFFDCA9
981CED090EDA9EE6AA94E39ECA4C5B6D45B092ACE92E0619B48F93E8D97484D05E1
FE1D692B90572F4D3826B0BC31E7FE6F3A48B759D325CDBA0A68F0F41DD3B',
 'public_key_int': 196903632939704349821326511449682872949973502109
841447081216670921083616934880886021118366574969409054737845252407062908586498910822966808907165210495345675110172359502170428207661759482903703114766546092487782120560822022251652795181706841287542903068625402445023618231936141602500769988309946179437763016741924528494923094784536637728127193048866530558458996043392685944574136516548253263771799146790636731606520650494024156302955354372458366987065649991698207604380194898606377716859545003006464839126627493612055558367306670537602581393311565379848265194029187871897807305698397770519742262220763895516557193946 19,
 'public_key_size': 2048,
 'secret_key': None,
 'secret_key_int': None,
 'secret_key_size': None,
 'module': '0X8AF90DE46B7ED7675DFB4FF39A315B3080E36A2ED2811D596FCCE
C3F23DAC69533EEF57225DEEFF492BBF70E63CA3FD2C02F926EC43137F0E6459FE6
DB31E0AF63B457FD4DD77366CE1249EA3479E7C5BD94A6CB1ED7CA22CA170C5F5B6
0C4459A63176375CCBC0F251AAC3CA9DBE992E4CC9782D2A2F2990A34CA6BD86BA9
CBE25F68B1F0B1CBAAA0D1E0BEB0717728796A9BDD1EF1F97DF63E6257FE7F57EE2
420D3F8C62F031B57E6B7124423520FD831C7757E91E039E141014905D6E74A5B88
256ABE607998B14C2B5440F8199C0F15191F8C3367D935D42A3FF3024350F41DE12
7C050D83E8A7FDFC9C322972E421091D19C1BAB418561FC319D2AD5E20A07B2868B
D667B43E02210CD9C8FFB926B11DC7B3F725B023098B04A6301EDD79AB93C1E21E9
2273908E7B8BDDC3424E145949C36ABD43886C5C493281B09C74C99998F570D343B
0152B1DD2FF115E7FB86A5D8E78DD5DE36FE197F9BBEA13D9E2FF9A4560A6FDEF2C
C80B1372578FCCB35511E448E364980AB2633471D19128649D62B01B86CC37DE81E
83C9F32D79FC69D5DB54995FD744CAA59000107C841DC2B74A2F912EA70A8D9BE4E
1DE2B858AF18003F2369A0870F83AFF2688E0F3970071AFA695D44CDD04F0DF6AAC
12EF06E86CD5DC91368BC736AEE25D05E0E1B6040B6C8EEFA4A38FFC4F045FD352D
76150E67F940205B0B67E1ADFED1202DF',
 'module_int': 566959837486774419959498647040211608577475022591376109566821842525045372777277327028692352107962888554788853865582578880933538415386786230715579915969688110440049689507358451607141341140693756454433019356535198108570930610343909554858053665449171299729689031390424109396672480488400844957328537852021912967236218706183420414924966054683070495260298534026986301927832315098053531576608111119625610099182834772940745025759653405995615466822147720927315822057842419914665974805552466815158175070336727050521027428866481811668328047120858633158227804838852683690679250043936263409003297829105054997330911308388482436086924798787825328708101561747351771320695834051571872474163371555244491430697417787370383539731031908705578277890487540578784542849551668075020456987729416120964008445049189679586422034663886746046989537851929328345486105799861452778872010944042096631202060026842867061313283182896586817548211275959753544097877310405889148594314083258954995335150848126715841224684435916789178678882082262619175815645021317176772192617176022393933462430392188561287738163561262200812313842703124283382109512274943908446354990419412154617011258885241015806773415086993886093741965104741823489497474352960229610444278693733360220096437 43,
 'block_size': 4095,
 'block_size_full': 4096}

In [7]:
```
# Создам отдельный объект для расштфрования и
# передам ему закрытый ключ

decryptor = RSA(secret_key_path="key.secret")
decryptor
```

Out[7]:
```
<rsa.RSA at 0x7f69c8c717d0>
```

In [8]:
```
# Также пристально посмотрим на объект через специальный метод
# Этот объект напротив - содержит лишь ключи для расшифрования
# Таким образом можно ограничить функциональность класса просто
# передав или не передавая ей нужные данные.

decryptor.__str__()
```

Out[8]:

```
{'public_key': None,
 'public_key_int': None,
 'public_key_size': None,
 'secret_key': '0X15971E2D9CA67D18B9B9D8F722F5744C04BDAA4FF8A5C22B3
63FB2CFEF8E2B7A19F5BD744B6726853FBFAAED45739CBE55AC25C3B2C37A8F9A99
488604447389BF6AB651BB26CAE65B374921062994385A0FC7F05FA6EFD6F5CB898
E28F0D12E5C93C2CC5874585568E8618F1C9EAB658A2F86DDE7A12E13AD82A55CA3
9B1D404D9C0139151AD46A313F5A4BB996D91B22F50234CFCD0E9FCAC6F9FFA38F3
9040C8974ABD70D744099D96DDB1E61AC4DF80CC7BD8ADAF774D132707C5CBC38C1
AC340F5521A88B21D0E330399B398DF7BBDBD70D6BFF036CD6B5013AA4148CEDA35
23FFE9EF71117910A502D8B85582AB24FDCFAEEA22AC8899F1DE90C5D346E66AE71
655F71945CFC6FC153A668A5B8C60E58B33A0EB86B42F8548130BC94CA82B28A0B1
B11F187A47C2DB737F6E6B309BCA306894AE54BBD81F6196500A12FE06D85527832
5DB5F51498F651CC558FA5AFB2615AD0847FDFA3AD30375AE0FB26E1208B6128ED6
25EF34D4520F61EE4E42EF7D602FDB89CCF5E5E70B2A03A725EEAE43510B9E64B07
47C9B669014A0BE66ED648F6AAA6ECCE91B5F63DB61CDE335D6D9F8F0762967C72E
A820DBECBE01B8C3F534F76EDD9FFE5F81846737FBEF4A78DEB6ED25A322588E3A7
544C775B783E460A4EC721FB81DDC439C67E6BCED7C73EDF1F41DBDF11535BA533E
EC35B9B698B0E267582FAA8054CABD28FFDF3',
 'secret_key_int': 88080756588272392110640303024286031075402406586 0
690369261976813366997372642669620930594936521422971621376987517323
786427139773428447886420589853485645132475246928981044873048537539 5
685303024493823170575593766547834638350345946021041197583220540116 1
645545528059400331861480520014170978985389657234015111354387768337 9
397969672584016862044846307616045661507561360199572520955970961558 4
200467036558978466115976021222654571815621217293371613262204977319 9
466728590925125018261120895154967528672469560284502981628728463665 0
669629406485126202629660540819824178189152108564042598726090111776 3
583209189140194219878978223606971420142772201425229747388929823522 2
048958411932071125344078665840468978541038066618930214232569488373 3
263572881596415543273549829783780056066408488399683407628413208779 5
665546642857203404443058041834656179299097243048247231229756321722 4
786239600183033297086050110284880400591127946602766378178359823432 8
237783402509377893576917798144116679518465482834937803399413526058 6
323699690051572082893918103244434770579570360448904234441909416712 6
557275536535017263345927091779615194549980362032711631680197576844 9
746724424142463568727649576893396435974468721520091266010212446604 2
014051347408886497438620887651831528364703219,
 'secret_key_size': 4093,
 'module': '0X8AF90DE46B7ED7675DFB4FF39A315B3080E36A2ED2811D596FCCE
C3F23DAC69533EEF57225DEEFF492BBF70E63CA3FD2C02F926EC43137F0E6459FE6
DB31E0AF63B457FD4DD77366CE1249EA3479E7C5BD94A6CB1ED7CA22CA170C5F5B6
0C4459A63176375CCBC0F251AAC3CA9DBE992E4CC9782D2A2F2990A34CA6BD86BA9
CBE25F68B1F0B1CBAAA0D1E0BEB0717728796A9BDD1EF1F97DF63E6257FE7F57EE2
420D3F8C62F031B57E6B7124423520FD831C7757E91E039E141014905D6E74A5B88
256ABE607998B14C2B5440F8199C0F15191F8C3367D935D42A3FF3024350F41DE12
7C050D83E8A7FDFC9C322972E421091D19C1BAB418561FC319D2AD5E20A07B2868B
D667B43E02210CD9C8FFB926B11DC7B3F725B023098B04A6301EDD79AB93C1E21E9
2273908E7B8BDDC3424E145949C36ABD43886C5C493281B09C74C99998F570D343B
0152B1DD2FF115E7FB86A5D8E78DD5DE36FE197F9BBEA13D9E2FF9A4560A6FDEF2C
C80B1372578FCCB35511E448E364980AB2633471D19128649D62B01B86CC37DE81E
83C9F32D79FC69D5DB54995FD744CAA59000107C841DC2B74A2F912EA70A8D9BE4E
1DE2B858AF18003F2369A0870F83AFF2688E0F3970071AFA695D44CDD04F0DF6AAC
12EF06E86CD5DC91368BC736AEE25D05E0E1B6040B6C8EEFA4A38FFC4F045FD352D
76150E67F940205B0B67E1ADFED1202DF',
 'module_int': 566959837486774419959498647040211608577475022591376 1
095668218425250453727772773287028692352107962888554788853865582578 8
809335384153867862307155799159696881104400496895073584516071413411 4
069375645443301935653519810857093061034390955485805366544917129972 9
689031390424109396672480488400844957328537852021912967236218706183 4
204149249660546830704952602985340269863019278323150980535315766081 1
```

```
11196256100991828347729407450257596534059956154668221477209273158220578424199146659748055524668151581750703367270505210274288664818116683280471208586331582278048388526836906792500439362634090032978291050549973309113083884824360869247987878253287081015617473517713206958340515718724741633715552444914306974177873703835397310319087055782778904875405787845428495516680750204569877294161209640084450491896795864220346638867460469895378519293283454861057998614527788720109440420966312020600268428670613132831828965868175482112759597535440978773104058891485943140832589549953351508481267158412246844359167891786788820822626191758156450213171767721926171760223939334624303921885612877381635612622008123138427031242833821095122749439084463549904194121546170112588852410158067734150869938860937419651047418234894974743529602296104442786937336022009643743,
 'block_size': 4095,
 'block_size_full': 4096}
```

In [9]:
```
# Хотя можно было все сделать одним объектов
# просто передав ему оба ключа

rsa = RSA(public_key_path="key.public", secret_key_path="key.secret"
rsa.__str__()
```

Out[9]:
{'public_key': '0X9BFA4C04A772DD0CD052F930680FB65DC1AB85954C4284DDF
9571AD8EBEBBEA3F82DC6B4CA1FDECA0A033941F40537E453550B41A3C241BD0D2A
8C07C9E518EFF90E6CCA816FACD93EDD8E178631270037BB6D4162472BF123FB0C1
10A8F40367B934D51EF85E94E0FD2B2AF71CF119A78633F95E6C64F832A150A5CE7
C323CC7E32B25279F7D5329FA6E01E7BF87AB17AE53E78B46E832CB9E547667EF74
783D2CEBD239C9B47B1AB6999BFFB9415F1959B018E0A6288003C1CF0945AFFDCA9
981CED090EDA9EE6AA94E39ECA4C5B6D45B092ACE92E0619B48F93E8D97484D05E1
FE1D692B90572F4D3826B0BC31E7FE6F3A48B759D325CDBA0A68F0F41DD3B',
 'public_key_int': 196903632939704349821326511449682872949973502109
841447081216670921083616934880886021118366574969409054737845252407062908586498910822966808907165210495345675110172359502170428207661759482903703114766546092487782120560822022251652795181706841287542903068625402445023618231936141602500769988309946179437763016741924528494923094784536637728127193048866530558458996043392685944574136516548253263771799146790636731606520650494024156302955354372458366987065649991698207604380194898606377716859545003006464839126627493612055558367306670537602581393311565379848265194029187871897807305698397770519742262220763895516557193946 19,
 'public_key_size': 2048,
 'secret_key': '0X15971E2D9CA67D18B9B9D8F722F5744C04BDAA4FF8A5C22B363FB2CFEF8E2B7A19F5BD744B6726853FBFAAED45739CBE55AC25C3B2C37A8F9A99488604447389BF6AB651BB26CAE65B374921062994385A0FC7F05FA6EFD6F5CB898E28F0D12E5C93C2CC5874585568E8618F1C9EAB658A2F86DDE7A12E13AD82A55CA39B1D404D9C0139151AD46A313F5A4BB996D91B22F50234CFCD0E9FCAC6F9FFA38F39040C8974ABD70D744099D96DDB1E61AC4DF80CC7BD8ADAF774D132707C5CBC38C1AC340F5521A88B21D0E330399B398DF7BBDBD70D6BFF036CD6B5013AA4148CEDA3523FFE9EF71117910A502D8B85582AB24FDCFAEEA22AC8899F1DE90C5D346E66AE71655F71945CFC6FC153A668A5B8C60E58B33A0EB86B42F8548130BC94CA82B28A0B1B11F187A47C2DB737F6E6B309BCA306894AE54BBD81F6196500A12FE06D855278325DB5F51498F651CC558FA5AFB2615AD0847FDFA3AD30375AE0FB26E1208B6128ED625EF34D4520F61EE4E42EF7D602FDB89CCF5E5E70B2A03A725EEAE43510B9E64B0747C9B669014A0BE66ED648F6AAA6ECCE91B5F63DB61CDE335D6D9F8F0762967C72EA820DBECBE01B8C3F534F76EDD9FFE5F81846737FBEF4A78DEB6ED25A322588E3A7544C775B783E460A4EC721FB81DDC439C67E6BCED7C73EDF1F41DBDF11535BA533EEC35B9B698B0E267582FAA8054CABD28FFDF3',
 'secret_key_int': 880807565882723921106403030242860310754024065860069036926197681336699737264266962093059493652142297162137698751732378642713977342844788642058985348564513247524692898104487304853753956853030244938231705755937665478346383503459460210411975832205401161645545528059400331861480520014170978985389657234015111354387768337939769672584016862044846307616045661507561360199572520955970961558420046703655897846611597602122265457181562121729337161326220497731994667285909251250182611208951549675286724695602845029816287284636650669629406485126202629660540819824178189152108564042598726090111776358320918914019421987897822360697142014277220142522974738892982352220489584119320711253440786658404689785410380666189302142325694883733263572881596415543273549829783780056066408488399683407628413208779566554664285720340444305804183465617929909724304824723122975632172247862396001830332970860501102848804005911279466027663781783598234328237783402509377893576917798144116679518465482834937803399413526058632369969005157208289391810324443477057957036044890423444190941671265572755365350172633459270917796151945499803620327116316801975768449746724424142463568727649576893396435974468721520091266010212446604201405134740888649743862088765183152836470321 9,
 'secret_key_size': 4093,
 'module': '0X8AF90DE46B7ED7675DFB4FF39A315B3080E36A2ED2811D596FCCEC3F23DAC69533EEF57225DEEFF492BBF70E63CA3FD2C02F926EC43137F0E6459FE6DB31E0AF63B457FD4DD77366CE1249EA3479E7C5BD94A6CB1ED7CA22CA170C5F5B60C4459A63176375CCBC0F251AAC3CA9DBE992E4CC9782D2A2F2990A34CA6BD86BA9CBE25F68B1F0B1CBAAA0D1E0BEB0717728796A9BDD1EF1F97DF63E6257FE7F57EE2420D3F8C62F031B57E6B7124423520FD831C7757E91E039E141014905D6E74A5B88

```
256ABE607998B14C2B5440F8199C0F15191F8C3367D935D42A3FF3024350F41DE12
7C050D83E8A7FDFC9C322972E421091D19C1BAB418561FC319D2AD5E20A07B2868B
D667B43E02210CD9C8FFB926B11DC7B3F725B023098B04A6301EDD79AB93C1E21E9
2273908E7B8BDDC3424E145949C36ABD43886C5C493281B09C74C99998F570D343B
0152B1DD2FF115E7FB86A5D8E78DD5DE36FE197F9BBEA13D9E2FF9A4560A6FDEF2C
C80B1372578FCCB35511E448E364980AB2633471D19128649D62B01B86CC37DE81E
83C9F32D79FC69D5DB54995FD744CAA59000107C841DC2B74A2F912EA70A8D9BE4E
1DE2B858AF18003F2369A0870F83AFF2688E0F3970071AFA695D44CDD04F0DF6AAC
12EF06E86CD5DC91368BC736AEE25D05E0E1B6040B6C8EEFA4A38FFC4F045FD352D
76150E67F940205B0B67E1ADFED1202DF',
 'module_int': 566959837486774419959498647040211608577475022591376109566821842525045372777277327028702869235210796288855478885386558257888093353841538678623071557991596968811044004968950735845160714134114069375645443301935653519810857093061034390955485805366544917129972968903139042410939667248048840084495732853785202191296723621870618342041492496605468307049526029853402698630192783231509805353157660811111962561009918283477294074502575965340599561546682214772092731582205784241991466597480555246681515817507033672705052102742886648181166832804712085863315822780483885268369067925004393626340900329782910505499733091130838848243608692479878782532870810156174735177132069583405157187247416337155524449143069741778737038353973103190870557827789048754057878454284955166807502045698772941612096400844504918967958642203466388674604698953785192932834548610579986145277887201094404209663120206002684286706131328318289658681754821127595975354409787731040588914859431408325895499533515084812671584122468443591678917867888208226261917581564502131717677219261717602239393346243039218856128773816356126220081231384270312428338210951227494390844635499041941215461701125888524101580677341508699388609374196510474182348949747435296022961044427869373336022009643743,
 'block_size': 4095,
 'block_size_full': 4096}
```

In [10]:
```
# Проверим правильность работы ключей у объектов encrypted
# и decrypted. Сгенерируем случайное число и зашифруем а
# затем расшифруем их

import random

_int = random.randrange(2, 100)

encr = pow(_int, encryptor.public_key_int, encryptor.module_int)
decr = pow(encr, decryptor.secret_key_int, decryptor.module_int)

bool(decr == _int)
```

Out[10]: True

In [11]:
```
# Проделаем то же самое с объектом rsa чтобы подтвердить
# корректность суждений

_int = random.randrange(2, 100)

encr = pow(_int, rsa.public_key_int, rsa.module_int)
decr = pow(encr, rsa.secret_key_int, rsa.module_int)

bool(decr == _int)
```

Out[11]: True

Часть 2. Зашифрование и расширование

In [12]:
```
# Для демонстрации зашифрования и расшифрования мною были
# подготовлены два файла с тескстом open.txt и изображением
# image.png

# Передам методу encrypt относительный путь к файлу для
# зашифрования и название файлы куда дб записан зашифрованый
# файл encrypted.txt

rsa.encrypt("open.txt", "encrypted.txt")
```

In [13]:
```
# Попробуем прочитать зашифрованый файл

!cat encrypted.txt
```

```
g♪⍉A W7L$W)ʏ'@3♪2A x
♫:Q<⌐F^⍉!ZQHg|Ę[Dv♪l9xL∝E
ϟ♀U[<MhfkH62►.♪≥▷⌀]rL D
f_∝Uj
[⌀DH♂<hwiŸX2~c+c=@3'∴?f`|
¤,xPsk|H▷◄1.∝,s(^]►`c
+8+NpG►1♀'=♀fb A&☎?9ɾ`|F99
♀#ijgi0.(q_5GP▷T►◷j*\z
2∴⌐柔%iɲ6ʌvǰ~19y[5 .H ℵAN;∴⌐O
Ԗx Ԍ q⌐-*tV`ħu7`y 9 9。~e
⌐yIف|Kpp[;>f*►V`⌐!i◁dgs(aϕ
s5 R
```

In [14]:
```
# Передам методу decrypt относительный путь к файлу для
# расшифрования и название файлы куда дб записан расшифрованый
# файл decrypted.txt

rsa.decrypt("encrypted.txt", "decrypted.txt")
```

In [15]:
```
!cat decrypted.txt
```

```
Прощай немытая Россия,
Страна рабов, страна господ,
И вы, мундиры голубые,
И ты, им преданный народ.

Быть может, за стеной Кавказа
Сокроюсь от твоих пашей,
От их всевидящего глаза,
От их всеслышащих ушей.

Михаил Лермонтов, 1841г.
```

In [16]:
```
# Проверим корректность расшифровки

with open("open.txt", "rb") as file:
    open_text = file.read()

with open("decrypted.txt", "rb") as file:
    decrypted_text = file.read()

bool(open_text == decrypted_text)
```

Out[16]:
```
True
```

In [17]:
```
# Установлю библиотеку для просмотра изображений

!pip3 install pillow
```

```
Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: pillow in /usr/lib/python3/dist-pack
ages (9.4.0)
```

In [18]:
```
# Зашифрую изображение и сохраню в файл encrypted.png

rsa.encrypt("image.png", "encrypted.png")
!ls -l
```

```
total 1212
-rw-r--r-- 1 evgeny evgeny 432799 Apr 25 17:03 analyze.ipynb
-rw-r--r-- 1 evgeny evgeny    408 Apr 25 17:05 decrypted.txt
-rw-r--r-- 1 evgeny evgeny  88091 Apr 25 17:05 demo.ipynb
-rw-r--r-- 1 evgeny evgeny 287142 Apr 23 17:18 demo.pdf
-rw-r--r-- 1 evgeny evgeny  35328 Apr 25 17:05 encrypted.png
-rw-r--r-- 1 evgeny evgeny    512 Apr 25 17:05 encrypted.txt
-rw-r--r-- 1 evgeny evgeny  35117 Apr 23 16:57 image.png
-rw-r--r-- 1 evgeny evgeny   4222 Apr 24 12:50 keygen.py
-rw-r--r-- 1 evgeny evgeny   1541 Apr 25 17:05 key.public
-rw-r--r-- 1 evgeny evgeny   2053 Apr 25 17:05 key.secret
-rw-r--r-- 1 evgeny evgeny 306562 Apr 25 16:42 mayday.png
-rw-r--r-- 1 evgeny evgeny    408 Apr 20 15:36 open.txt
drwxr-xr-x 2 evgeny evgeny   4096 Apr 24 12:51 __pycache__
-rw-r--r-- 1 evgeny evgeny   5160 Apr 23 16:57 rsa.py
drwxr-xr-x 6 evgeny evgeny   4096 Apr 15 15:18 venv
```

In [19]:
```
# Попробую открыть файл

from PIL import Image

try:
    Image.open("encrypted.png")
except:
    print("Невозможно прочитать файл!")
```

```
Невозможно прочитать файл!
```

In [20]:
```
# Расшифрую изображение и сохраню в файл decrypted.png

rsa.decrypt("encrypted.png", "decrypted.png")
!ls -l
```

```
total 1248
-rw-r--r-- 1 evgeny evgeny 432799 Apr 25 17:03 analyze.ipynb
-rw-r--r-- 1 evgeny evgeny  35117 Apr 25 17:06 decrypted.png
-rw-r--r-- 1 evgeny evgeny    408 Apr 25 17:05 decrypted.txt
-rw-r--r-- 1 evgeny evgeny  88091 Apr 25 17:05 demo.ipynb
-rw-r--r-- 1 evgeny evgeny 287142 Apr 23 17:18 demo.pdf
-rw-r--r-- 1 evgeny evgeny  35328 Apr 25 17:05 encrypted.png
-rw-r--r-- 1 evgeny evgeny    512 Apr 25 17:05 encrypted.txt
-rw-r--r-- 1 evgeny evgeny  35117 Apr 23 16:57 image.png
-rw-r--r-- 1 evgeny evgeny   4222 Apr 24 12:50 keygen.py
-rw-r--r-- 1 evgeny evgeny   1541 Apr 25 17:05 key.public
-rw-r--r-- 1 evgeny evgeny   2053 Apr 25 17:05 key.secret
-rw-r--r-- 1 evgeny evgeny 306562 Apr 25 16:42 mayday.png
-rw-r--r-- 1 evgeny evgeny    408 Apr 20 15:36 open.txt
drwxr-xr-x 2 evgeny evgeny   4096 Apr 24 12:51 __pycache__
-rw-r--r-- 1 evgeny evgeny   5160 Apr 23 16:57 rsa.py
drwxr-xr-x 6 evgeny evgeny   4096 Apr 15 15:18 venv
```

In [21]:
```
# Проверим корректность расшифровки

with open("image.png", "rb") as file:
    source_image = file.read()

with open("decrypted.png", "rb") as file:
    decrypted_image = file.read()

bool(source_image == decrypted_image)
```

Out[21]:
```
True
```

In [22]:
```
# Попробую открыть расшифрованый файл

Image.open("decrypted.png")
```

Out[22]:

Спасибо за внимание и да пребудет с вами сила!