Getting started

```
cd src
pip3 install pycryptodome
./test.sh
./demo.sh
```

Pycryptodome is a Python package implementing AES ciphers. We only use it in cipher.py.

Directory structure

```
.
├── README.md
├── report.pdf
└── src
    ├── bbs.py
    ├── cipher.py
    ├── demo.sh
    ├── files
    │   ├── 2048-bit MODP Group
    │   │   ├── generator.txt
    │   │   ├── prime.txt
    │   │   ├── test_xA.txt
    │   │   ├── test_xB.txt
    │   │   ├── test_yA.txt
    │   │   ├── test_yB.txt
    │   │   └── test_Z.txt
    │   └── really_secret_file.txt
    ├── keygen.py
    └── test.sh
```

Files

- README.md : this file
- report.pdf : report of the assignment
- src/ : the source code directory
    - bbs.py : Python implementation of the Blum Blum Shub PRNG
    - cipher.py : Python implementation of AES encryption/decryption
    - keygen.py : Python implementation of the Diffie-Hellman key exchange scheme

    - files/ : a directory containing files used by the different tools
        - really_secret_file.txt : the file Alice wishes to send to Bob without disclosing its contents
        - 2048-bit MODP Group/ : the elements of a Diffie-Hellman group defined by IETF standards (https://tools.ietf.org/html/rfc5114)
    - test.sh : a shell script running implementation tests
    - demo.sh : a shell script providing a usecase example of how to use the tools implemented

User manual

All these manuals can be found using `python3 script_name.py --help`. To see usecase usage of these tools, take a look at the commands used in demo.sh.

Diffie-Hellman manual

```
$ python3 keygen.py --help

usage: keygen.py [-h] --mode {generate,merge,test} [--prime PRIME]
                 [--root ROOT] [--secret SECRET] [--verbose] [--output
OUTPUT]
                 [--public PUBLIC]

Generate public and private keys with the Diffie-Hellmann algorithm

optional arguments:
  -h, --help            show this help message and exit
  --mode {generate,merge,test}
                        Generate a public key, compute a shared private
key,
                        or test program
  --prime PRIME         Prime used (hex or decimal) for key generation
  --root ROOT           Primitive root (hex or decimal) used for key
                        generation
  --secret SECRET       Private key (hex or decimal) used for key
generation
  --verbose             Display parameters used for key generation
  --output OUTPUT       File to which the public key is written (standard
                        ouput if not specified)
  --public PUBLIC       Public key (hex or decimal) to be merged with the
                        private key
```

Blum Blum Shub manual

```
$ python3 bbs.py --help

usage: bbs.py [-h] --seed SEED [--size SIZE] [--output OUTPUT] [--verbose]
```

```
Generate a random number using Blum Blum Shub algorithm

optional arguments:
  -h, --help       show this help message and exit
  --seed SEED      Seed used for random number generation
  --size SIZE      Size in bits of the generated number, 128 if not
specified
  --output OUTPUT  File to which the random number is written
  --verbose        Display parameters used for key generation
```

AES manual

```
$ python3 cipher.py --help

usage: cipher.py [-h] --mode {encrypt,decrypt} --key KEY --input INPUT
                 [--output OUTPUT] [--verbose]

Encrypt and decrypt data using AES

optional arguments:
  -h, --help            show this help message and exit
  --mode {encrypt,decrypt}
                        Encrypt or decrypt data
  --key KEY             The key used for encryption or decryption
  --input INPUT         Path to the file to encrypt or decrypt
  --output OUTPUT       Path to wich the encrypted or decrypted data is
                        written. If not specified, output is redirected to
                        stdout
  --verbose             Run in verbose mode
```

Requirements

- Python 3.6 or above
- Pip 9.0.1 or above