Getting started

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

Pycryptodome is a Python package implementing AES ciphers used it in `cipher.py`. Pillow is an image library used to create bitmaps of the random numbers generated by `bbs.py`.

Directory structure

```
.
├── README.pdf
├── report.pdf
└── src
    ├── files
    │   ├── 2048-bit MODP Group
    │   ├── AES_test_data
    │   └── really_secret_file.txt
    ├── bbs.py
    ├── cipher.py
    ├── demo.sh
    ├── keygen.py
    └── test.sh
```

Files

- `README.pdf` : 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 compliant with IETF standards found in this RFC
        - `AES test data/` : plaintext and ciphertext equivalents from NIST example values
    - `test.sh` : a shell script running implementation tests
    - `demo.sh` : a shell script providing a use case example of how to use the tools implemented together

User manual

All these manuals can be found using `python3 script_name.py --help`. To see use case 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
                        output 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] --mode {generate,test} --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
  --mode {generate,test}
                        Generate a random number or test randomness
  --seed SEED           Seed (hex or decimal) used for random number
                        generation
  --size SIZE           Size in bits of the generated number, 128 if not
                        specified (use 128, 192 or 256 for AES
compatibility)
  --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,test} [--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,test}
                        Encrypt data, decrypt data, or run the tests
  --key KEY             The key used for encryption or decryption
  --input INPUT         Path to the file to encrypt or decrypt
  --output OUTPUT       Path to which 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