Class Activity 1:

```
#1. # Python 3 code to check
# available algorithms

import hashlib

# prints all available algorithms
print ("The available algorithms are : ", end ="")
print (hashlib.algorithms_guaranteed)


# 2. Import sha256 from the hashlib Python library
from hashlib import sha256

#3. Create a variable called text. Initialise the variable with this
string I am excited to learn about blockchain
text = "I am excited to learn about blockchain!"

#4. Create a sha256 hash object, using the constructor sha256() and
pass the text variable as its argument. Assign the value of this object
to a variable called hash_result.Be sure to use the .encode() method on
the text variable. Call the .hexdigest() method on hash_result and
print the result

# encoding I am excited to learn about blockchain using encode()
# then sending to SHA256()

hash_result = hashlib.sha256(text.encode())

# printing the equivalent hexadecimal value.
print("The hexadecimal equivalent of SHA256 is : ")
print(hash_result.hexdigest())

#\r moves the cursor to the beginning of the line and then outputting
characters as normal(carriage return)
print ("\r")

# Try changing the text variable and see how the hash gets completely
different.

# initialising string
```

```
str = "jetlearn"

# encoding jetlearn using encode()
# then sending to SHA224()
result = hashlib.sha224(str.encode())

# printing the equivalent hexadecimal value.
print("The hexadecimal equivalent of SHA224 is : ")
print(result.hexdigest())

print ("\r")

# initialising string
str = "GeeksforGeeks"

# encoding GeeksforGeeks using encode()
# then sending to SHA512()
result = hashlib.sha512(str.encode())

# printing the equivalent hexadecimal value.
print("The hexadecimal equivalent of SHA512 is : ")
print(result.hexdigest())

print ("\r")

# initialising string
str = "Jetlearn"

# encoding Jetlearn using encode()
# then sending to SHA1()
result = hashlib.sha1(str.encode())

# printing the equivalent hexadecimal value.
print("The hexadecimal equivalent of SHA1 is : ")
print(result.hexdigest())
```

Class Activity 2: To implement something similar using the concepts of class, object and dictionary

```
import hashlib
import json
from time import time

chain = []


def block(proof, previous_hash=None):
    transaction = {
        'sender': 'Satoshi',
        'recipient': 'Mike',
        'amount': '5 ETH'
    }
    data = {
        'index': 1,
        'timestamp': time(),
        'transactions': transaction,
        'gas/fee': 0.1,
        'proof': proof,
        'previous_hash': previous_hash,
    }
    chain.append(data)
    print("block information:", data)
    string_object = json.dumps(data)
    block_string = string_object.encode()

    raw_hash = hashlib.sha256(block_string)
    hex_hash = raw_hash.hexdigest()
    print("Hash code of block:", hex_hash)


block(previous_hash="No previous Hash. Since this is the first block.",
proof=000)
```