1. Write a function that accepts a positive integer as a parameter and then returns a representation of that number in binary (base 2).

Hint: This is in many ways a trick question. Think!

Code:

```
def into_binary(n):
  return bin(n)[3:]
print(into_binary(10))
```

Output:

```
010
```

2. Write and test a function that takes an integer as its parameter and returns the factors of that integer. (A factor is an integer which can be multiplied by another to yield the original).

Code:

```
def find_factors(n):
  if n == 0:
    raise ValueError("Zero does not have factors.")
  n = abs(n)
  factors = []
  for i in range(1, int(n**0.5) + 1):
    if n % i == 0:
      factors.append(i)
      if i != n // i:
        factors.append(n // i)
  return sorted(factors)

num = int(input("Enter a number: "))
print(f"Factors of {num}: {find_factors(num)}")
```

Output:

```
Enter a number:  10
Factors of 10: [1, 2, 5, 10]
```

3. Write and test a function that determines if a given integer is a prime number. A prime number is an integer greater than 1 that cannot be produced by multiplying two other integers.

Code:

```
def is_prime(n):
  if n <= 1:
    return False
```

```
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

integer = int(input("Enter a number: "))
print(f"{integer} is prime: {is_prime(integer)}")
```

Output:

```
Enter a number:  10
10 is prime: False
```

4. Computers are commonly used in encryption. A very simple form of encryption (more accurately "obfuscation") would be to remove the spaces from a message and reverse the resulting string. Write, and test, a function that takes a string containing a message and "encrypts" it in this way.

Code:

```
def encrypt_message(message):
    return ''.join(message.replace(" ", "")[::-1])

user_message = input("Enter a message: ")
encrypted_message = encrypt_message(user_message)

print(f"Original Message: {user_message}")
print(f"Encrypted Message: {encrypted_message}")
```

Output:

```
Enter a message:  Apeal
Original Message: Apeal
Encrypted Message: laepA
```

5. Another way to hide a message is to include the letters that make it up within seemingly random text. The letters of the message might be every fifth character, for example. Write and test a function that does such encryption. It should randomly generate an interval (between 2 and 20), space the message out accordingly, and should fill the gaps with random letters. The function should return the encrypted message and the interval used. For example, if the message is "send cheese", the random interval is 2, and for clarity the random letters are not random:

send cheese

s e n d c h e e s e

sxyexynxydxy cxyhxyexyexysxye

Code:

```
import random
import string

# Function to encrypt the message
def encrypt(input_message):
    random_interval = random.randint(2, 20)
    result_message = ''
    for index, character in enumerate(input_message):
        result_message += character
        if index < len(input_message) - 1:
            result_message += ''.join(random.choices(string.ascii_lowercase, k=random_interval - 1))
    return result_message, random_interval

# Function to decrypt the message
def decrypt(encrypted_message, interval):
    decrypted_message = ''
    for i in range(0, len(encrypted_message), interval):
        decrypted_message += encrypted_message[i]
    return decrypted_message

original_message = "send cheese"
encrypted_output, interval_used = encrypt(original_message)

print(f"Original message: {original_message}")
print(f"Encrypted message: {encrypted_output}")
print(f"Interval used: {interval_used}")
```

Output:

```
Original message: send cheese
Encrypted message: syhklpntwwmpzrkggcpeoibzyooeqaniuhkucxnoffrwebxakksdoofgjdwcvqtegogmfyzoarpx qjvsippctnnpjleconcuh
npgmjrsdcnoccjpohwavwxhrbwczblfimdseusstqnlazmkqsgydbuecvadfakjtfwpylwtmmsdhfxmnnqafgfrbwpaae
Interval used: 19
```

6. Write a program that decrypts messages encoded as above.

Code:

```
import random
import string

# Function to encrypt the message
def encrypt(input_message):
    random_interval = random.randint(2, 20)
    result_message = ''
    for index, character in enumerate(input_message):
        result_message += character
        if index < len(input_message) - 1:
```

```
        result_message += ''.join(random.choices(string.ascii_lowercase, k=random_interval - 1))
    return result_message, random_interval

# Function to decrypt the message
def decrypt(encrypted_message, interval):
    decrypted_message = ''
    for i in range(0, len(encrypted_message), interval):
        decrypted_message += encrypted_message[i]
    return decrypted_message

original_message = "send cheese"
encrypted_output, interval_used = encrypt(original_message)

print(f"Original message: {original_message}")
print(f"Encrypted message: {encrypted_output}")
print(f"Interval used: {interval_used}")

# Decrypt the message
decrypted_message = decrypt(encrypted_output, interval_used)
print(f"Decrypted message: {decrypted_message}")
```

Output:

```
Original message: send cheese
Encrypted message: sjnuufvmhefeesrhcmqzwtplnogsnvaxqxludokurvpyucyv rgqrmhnmwpzcsrqbjuwwhkohfpinpzstolxeambdfxormenej
wgtgvumijashesoncpfioke
Interval used: 12
Decrypted message: send cheese
```