1. Ask the user for two integers named *larger* and *smaller*. Determine (and output) how many times larger can be halved while still be greater than smaller.

   Examples:

   - if *larger* = 1324 and *smaller* = 98, the result should be 3 since $1324 \rightarrow 662 \rightarrow 331 \rightarrow 165.5$
   - if *larger* = 624 and *smaller* = 8, the result should be 6 since
     $624 \rightarrow 312 \rightarrow 156 \rightarrow 78 \rightarrow 39 \rightarrow 19.5 \rightarrow 9.75$)

2. Write a program that asks the user for a word and then, using a loop, prints every other letter of the word starting with the second letter.

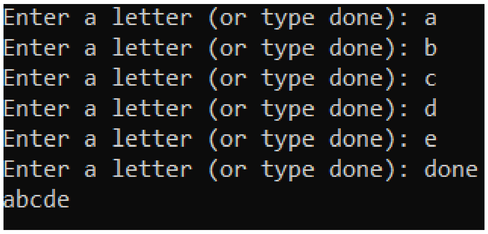
   Examples:

   - if user_word = ‟counterattack”, the result should be ‟oneatc”
   - if user_word = ‟banana sunday”, the result should be ‟aaasna”

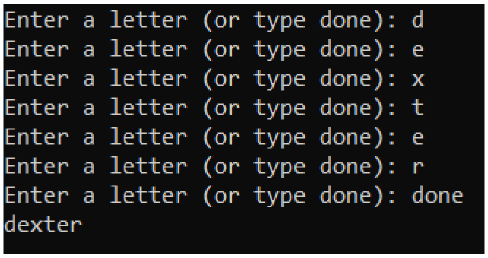
3. Using a loop, write a program that prints every even number between 37 and 1050 (inclusively).

4. Write a program to create a word one letter at a time. You should prompt the user to enter a single letter one at a time until they type *done*. Done they type done, output their newly created word.

   For example,



```
Enter a letter (or type done): a
Enter a letter (or type done): b
Enter a letter (or type done): c
Enter a letter (or type done): d
Enter a letter (or type done): e
Enter a letter (or type done): done
abcde
```

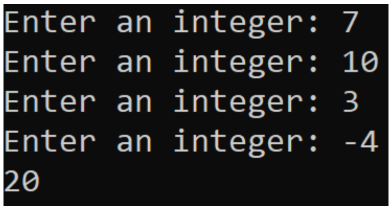


```
Enter a letter (or type done): d
Enter a letter (or type done): e
Enter a letter (or type done): x
Enter a letter (or type done): t
Enter a letter (or type done): e
Enter a letter (or type done): r
Enter a letter (or type done): done
dexter
```

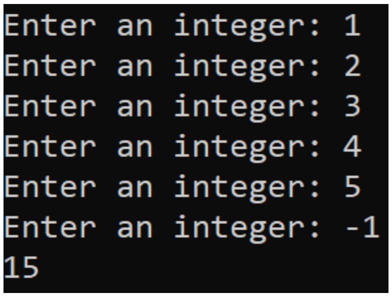
5. Using a loop, write code to calculate the sum of all odd numbers between 50 and 517. Print the result.

6. Write a program that repeatedly asks the user for integers until a negative integer is given.
   The program should keep track of the sum of the numbers and print the sum at the end
   (not including the negative number).

   For example,



```
Enter an integer: 7
Enter an integer: 10
Enter an integer: 3
Enter an integer: -4
20
```



```
Enter an integer: 1
Enter an integer: 2
Enter an integer: 3
Enter an integer: 4
Enter an integer: 5
Enter an integer: -1
15
```

7. Given a positive integer n, the following rules will always create a sequence that ends with 1, called Hailstone Sequence:

   (a) If n is even, divide by 2

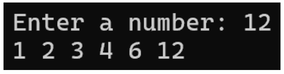
(b) If n is odd, multiply by 3 and add 1 (i.e. $3n + 1$)

(c) Continue until n is 1

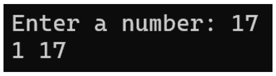
Write a program that prints the hailstone sequence starting at $n = 25$.

8. Write code that asks the user for an integer and then prints each number that is a factor of the input.

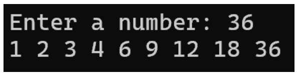
   For example,



```
Enter a number: 12
1 2 3 4 6 12
```



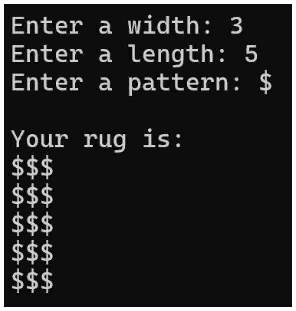
```
Enter a number: 17
1 17
```



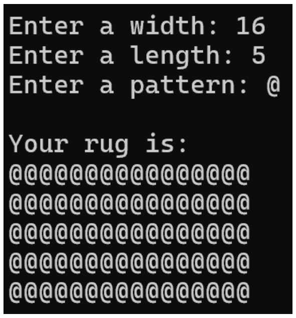
```
Enter a number: 36
1 2 3 4 6 9 12 18 36
```

9. You are the newest rug fashion designer on the scene, but you're running out of ideas. Write a program that will help you design rugs. The program should ask for a width, a length, and pattern, and then create a rug consisting of that pattern and dimensions.

   For example,



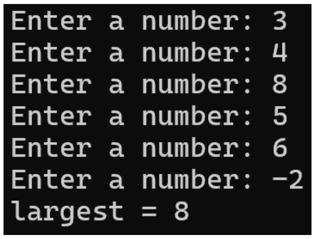
```
Enter a width: 3
Enter a length: 5
Enter a pattern: $

Your rug is:
$$$
$$$
$$$
$$$
$$$
```



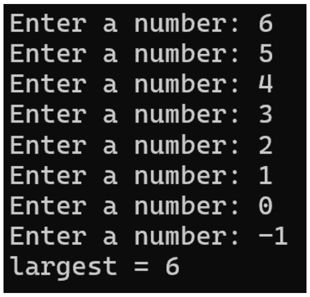
```
Enter a width: 16
Enter a length: 5
Enter a pattern: @

Your rug is:
@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@
```

10. Write a program that repeatedly asks the user for integers until a negative integer is given. Report back the largest **even** number the user entered (not including the negative number). If the user didn't enter any even numbers report back -1.

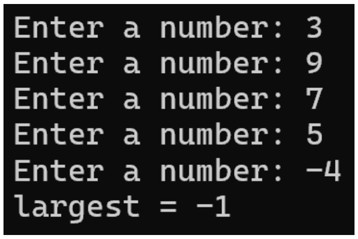
    For example,



```
Enter a number: 3
Enter a number: 4
Enter a number: 8
Enter a number: 5
Enter a number: 6
Enter a number: -2
largest = 8
```



```
Enter a number: 6
Enter a number: 5
Enter a number: 4
Enter a number: 3
Enter a number: 2
Enter a number: 1
Enter a number: 0
Enter a number: -1
largest = 6
```



```
Enter a number: 3
Enter a number: 9
Enter a number: 7
Enter a number: 5
Enter a number: -4
largest = -1
```

11. Write a program that asks the user for an integer. Calculate (and then print) the sum of all square numbers up to and including the user's number.

    For example,

    - if user_number = 3, the result should be 14 since $1^2 + 2^2 + 3^2 = 14$.
    - if user_number = 8, the result should be $1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + 8^2 = 204$.