Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns "driving the mouse." Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a "divide-and-conquer" strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

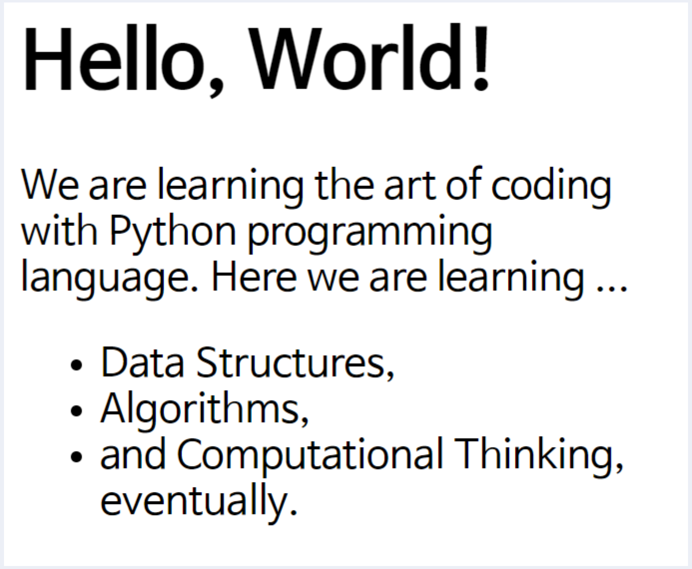
A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor’s question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students “divide and conquer.” When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Pair programming

Q1. The HTML document consists of many tags as shown below. Write a program that matches HTML document tags.

Matching HTML Tags

```
<html>
<body>
<h1>Hello, World!</h1>
<p> We are learning the art of coding
with Python programming language.
Here we are learning ... </p>
<ul>
<li> Data Structures, </li>
<li> Algorithms, </li>
<li> and Computational Thinking,
    eventually. </li>
</ul>
</body>
</html>
```

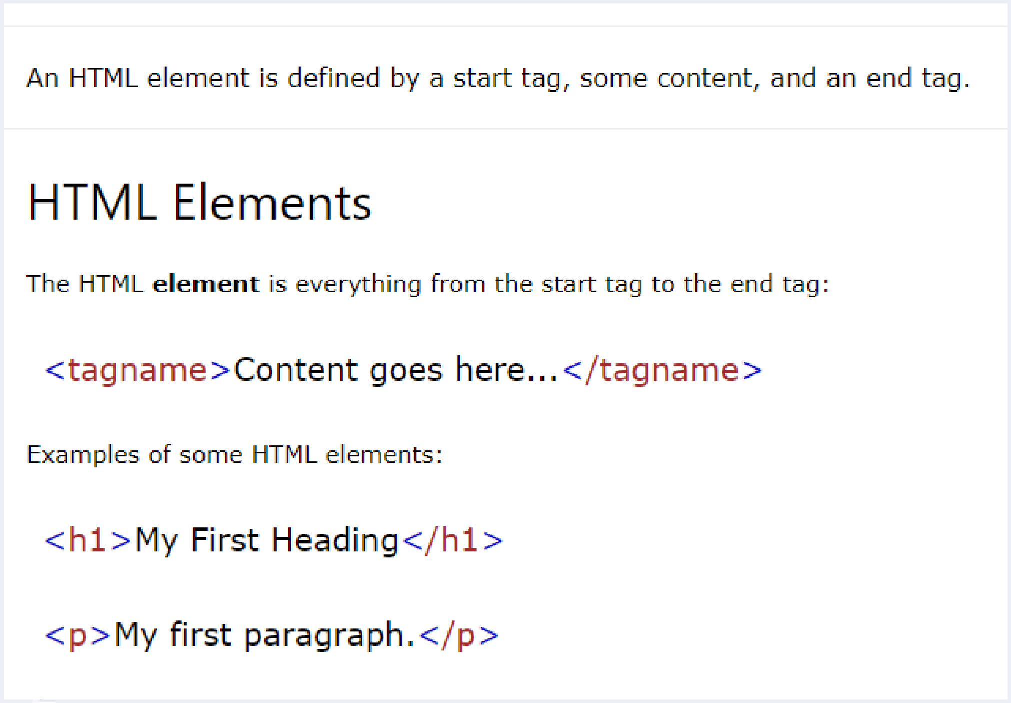


Hello, World!

We are learning the art of coding with Python programming language. Here we are learning ...

- Data Structures,
- Algorithms,
- and Computational Thinking, eventually.

Reference


An HTML element is defined by a start tag, some content, and an end tag.

HTML Elements

The HTML **element** is everything from the start tag to the end tag:

```
<tagname>Content goes here...</tagname>
```

Examples of some HTML elements:

```
<h1>My First Heading</h1>

<p>My first paragraph.</p>
```

https://www.w3schools.com/html/html_elements.asp

TIP

- In Python, the find() method is used to locate the start position of the substring. Find the ‘<‘ and ‘>’ to distinguish HTML document tags.
- If the distinguished tag includes ‘/,’ then it is an open tag, and if not, it is a close tag. Check the HTML tag matching in a similar way of using a stack to match parentheses.

```python
text = input("Input the string of HTML document: ")
```

```
Input the string of HTML document: <html> <body> <h1>Hello, World!</h1> <p> We are learning the art of coding
with Python programming language. Here we are learning ... </p>  <ul> <li> Data Structures, </li> <li> Algorit
hms, </li> <li> and Computational Thinking,      eventually. </li>  </ul> </body> </html>
```

```python
start = text.find('<')
while start != -1:
    end = text.find('>', start + 1)
    tag = text[start:end+1]
    print(tag, end = ' ')
    start = text.find('<', end + 1)
```

```
<html> <body> <h1> </h1> <p> </p> <ul> <li> </li> <li> </li> <li> </li> </ul> </body> </html>
```

Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor's question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students "divide and conquer." When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Pair programming

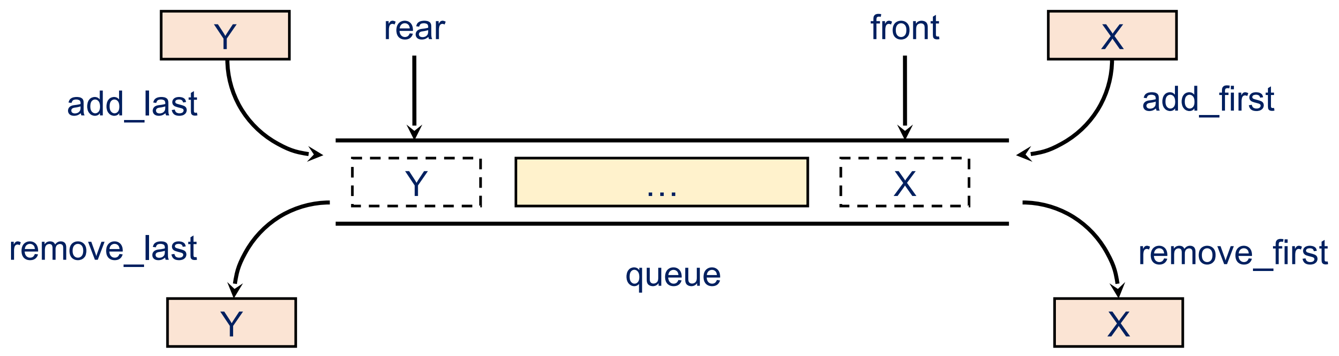
Q1. Refer to the class definition as follows to complete and test the Deque class.

```python
class Deque:

    def __init__(self):
        self.queue = []

    def add_first(self, item):
        ''' Add an item to the front of the queue '''

    def remove_first(self):
        ''' Remove and return the item from the front '''

    def add_first(self, item):
        ''' Add an item to the rear of the queue '''

    def remove_first(self):
        ''' Remove and return the item from the rear '''
```

Pair programming

TIP

- A queue is an abstract data type where only addition is possible on one side and only removal is possible on the other side. In contrast, the Double Ended Queue (Deque) is the data structure where addition and removal are possible on both sides.



Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns "driving the mouse." Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a "divide-and-conquer" strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

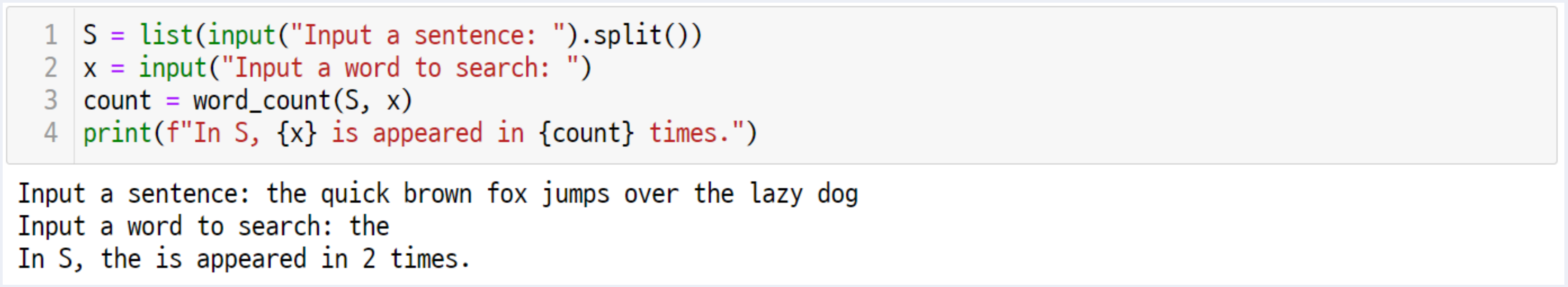
Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor's question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students "divide and conquer." When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Pair programming

Q1. Write a program to count how many times a specific word was used in the sentence entered by the user.

- In the sentence entered by the user, a word is distinguished by the presence of spacing.
- Use the input().split() function to create the list S that has each string as its element.
- Use the input() function to receive the word that will be searched by the user and save it to x.
- The word_count() function receives S and x as parameters and provides a return by counting how many x are present in the S.



```python
S = list(input("Input a sentence: ").split())
x = input("Input a word to search: ")
count = word_count(S, x)
print(f"In S, {x} is appeared in {count} times.")
```

```
Input a sentence: the quick brown fox jumps over the lazy dog
Input a word to search: the
In S, the is appeared in 2 times.
```

| Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns "driving the mouse." Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a "divide-and-conquer" strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

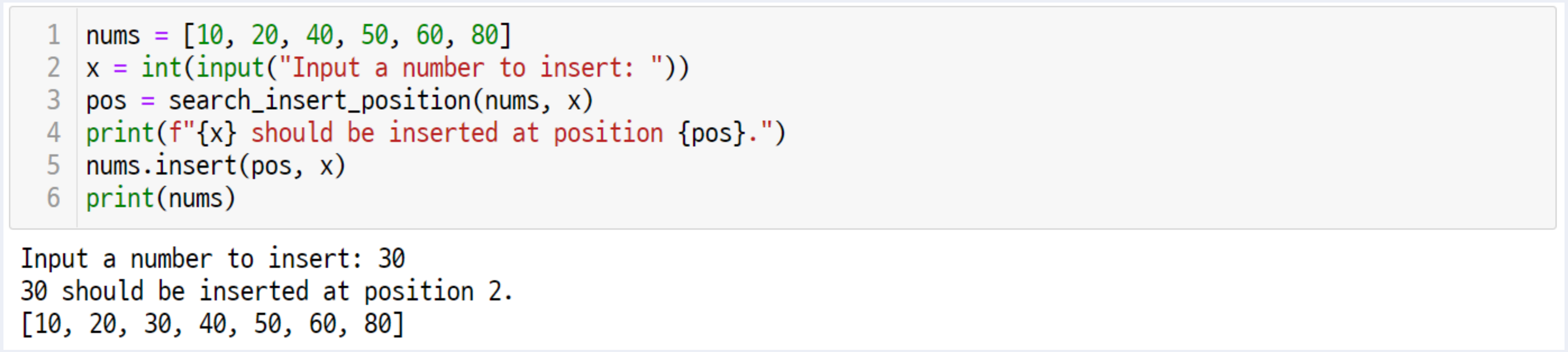
The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor’s question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students “divide and conquer.” When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Pair programming

Q1. If an ordered list ’nums’ and a random integer x are given, implement the function that returns the location of index where x will be inserted. However, the S should be an ordered list even after inserting x.



```python
nums = [10, 20, 40, 50, 60, 80]
x = int(input("Input a number to insert: "))
pos = search_insert_position(nums, x)
print(f"{x} should be inserted at position {pos}.")
nums.insert(pos, x)
print(nums)
```

```
Input a number to insert: 30
30 should be inserted at position 2.
[10, 20, 30, 40, 50, 60, 80]
```

Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns "driving the mouse." Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a "divide-and-conquer" strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

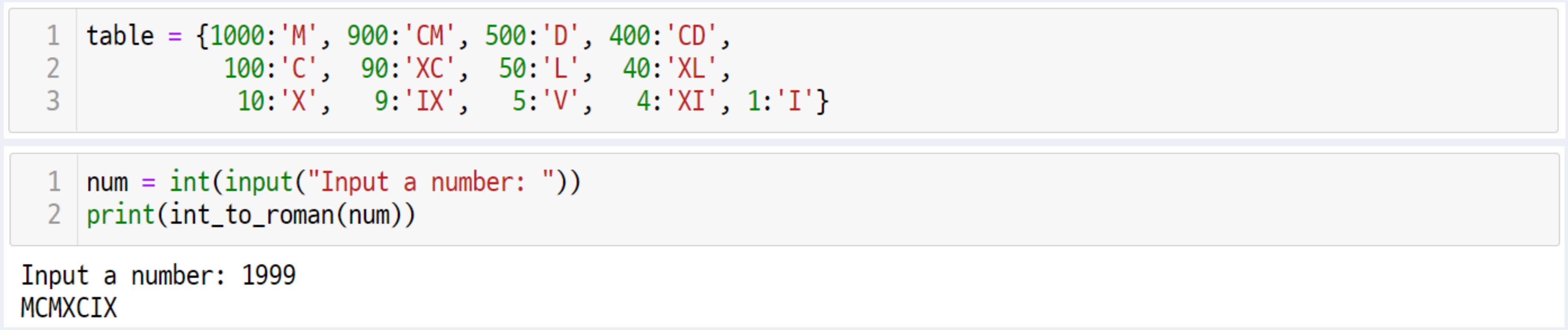
The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor's question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students "divide and conquer." When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Pair programming

Q1. Create a converter that converts Arabic numerals to Roman numerals by using the hash table provided below.



```python
table = {1000:'M', 900:'CM', 500:'D', 400:'CD',
           100:'C',  90:'XC',  50:'L',  40:'XL',
            10:'X',   9:'IX',   5:'V',   4:'XI', 1:'I'}
```

```python
num = int(input("Input a number: "))
print(int_to_roman(num))
```

```
Input a number: 1999
MCMXCIX
```

Examples of roman numbers:

369, 80, 29, 155, 14, 492, 348, 301, 469, 499

CCCLXIX, LXXX, XXIX, CLV, XIV, CDXCII, CCCXLVIII, CCCI, CDLXIX, CDXCIX,