```
#definin
g rules
            def get_rules():
                rules = {"Numbers":{
                                        "zero": 0,
                                        "one" : 1,
                                        "two": 2,
                                        "three": 3,
                                        "four": 4,
                                        "five": 5,
                                        "six": 6,
                                        "seven": 7,
                                        "eight": 8,
                                        "nine": 9,
                                        "ten": 10,
                                        "twenty": 20,
                                        "thirty": 30,
                                        "forty": 40,
                                        "fifty": 50,
                                        "sixty": 60,
                                        "seventy": 70,
                                        "eighty": 80,
                                        "ninety": 90,
                                        "hundred": 100
                                        },
                                "Tuples": {
                                            "single":1,
                                            "double":2,
                                            "triple":3,
                                            "quadruple":4,
                                            "quintuple":5,
                                            "sextuple":6,
```

```
                        "septuple":7,
                        "octuple":8,
                        "nonuple":9,
                        "decuple":10
                    },
            "General": {
                        "C M": "CM",
                        "P M": "PM",
                        "D M": "DM",
                        "A M": "AM"
                    }
            }
    return rules


#checking if word has comma at front or at last or at both  if true then return front,word and last
def check_front_last(word):
    front=""
    last=""
    if(len(word)>1):
        if word[-1]==',' or word[-1]=='.':
            last=word[-1]
            word=word[:-1]
        if word[0]==',' or word[0]=='.':
            front=word[0]
            word=word[1:]
    return front,word,last



#class for conversion
class SpokenToWritten:
```

```
    def __init__(self):


        self.rules=get_rules()
        self.paragraph=""
        self.ouptut_para=""


    #getting user input
    def get_user_input(self):


        self.paragraph=input("\n[IN]:Enter Your paragraph of spoken
english:\n\t")


        if not self.paragraph:
            raise ValueError("[Error]: You entered nothing.")


    #getting  user output
    def show_output(self):
        print("\n[OUT]:The input Spoken English Paragraph: \n\n \" "+
self.paragraph+"\"")
        print("\nConverted Written English Paragraph: \n\n \""
+self.ouptut_para+"\"")



    #main conversion function of spoken to written english
    def Convert(self):
        #splitting paragraph into individual words
        words_of_para=self.paragraph.split()


        #accessing defines rules
```

```
        numbers=self.rules['Numbers']
        tuples=self.rules['Tuples']
        general=self.rules['General']
        i=0
        no_of_words=len(words_of_para)
        #loop will run for the number of words in paragraph
        while i<no_of_words:

            front,word,last=check_front_last(words_of_para[i])
            #Word of paragraph may of form ',dollars.'
            if i+1!= no_of_words:
            #when word is of the form e.g.: two

front_n,next_word,last_n=check_front_last(words_of_para[i+1])
                if word.lower() in numbers.keys() and
(next_word.lower()=='dollars' or next_word.lower()=='dollar'):
                    self.ouptut_para=self.ouptut_para+"
"+front+"$"+str(numbers[word.lower()])+last
                    i=i+2


                elif word.lower() in tuples.keys() and len(next_word)==1:
                    #when word is of form Triple A
                    self.ouptut_para=self.ouptut_para+"
"+front_n+(next_word*tuples[word.lower()])+last_n
                    i=i+2
                elif (word+" "+next_word) in general.keys():
                    #if word is of form P M or C M
                    self.ouptut_para=self.ouptut_para+"
"+front+word+next_word+last_n
                    i=i+2
                else:
                    self.ouptut_para=self.ouptut_para+"
"+words_of_para[i]
                    i=i+1
            else:
```

```
                self.ouptut_para=self.ouptut_para+" "+words_of_para[i]
                i=i+1



#main function
def convert_sp_to_wr():
    #creating class object
    obj_spoken=SpokenToWritten()
    obj_spoken.get_user_input()
    obj_spoken.Convert()




    obj_spoken.show_output()
```