Algorithm

Create a global character array str to store entered string (of size 1000).

Function clear (char clear[ ])

Step 1: Function accepts a string as parameter
Step 2: Create a for loop to go from zero to last index of entered string, initializing all indexes with '\0'.
Step 3 : End

Function count Char ()

Step 1: Start
Step 2: display "\nCount \n......"
Step 3: store length of str in a variable n.
Step 4: Create 5 counter variables to store 5 category of characters (Lowercase, upper case, space, tabs, special character).
Step 5: Read string character by character using for loop and increase each counter as and when they occur in string.
Step 6: Special character also ~~is~~ has space in them so add space counter to special character.
Step 7: Print value stored in each counter.
Step 8: End.

Scanned with CamScanner

Function vandc()

```
Step 1 : start
Step 2 : Print the message "Vowels and Consonants"
Step 3 : Create an array of 26 characters
Step 4 : Run a for loop from i=0 to i<26
         and repeat step5 to step9 and do i++.
Step 5 : Store ASCII value of character in a
         variable ch.
Step 6 : If ch is upper case then do ch=ch-65
Step 7 : Else if ch is lower case then do
         ch=ch-97
Step 8 : else put ch=-1
Step 9 : If ch>=0 do c[ch]++
Step 10 : Print vowels on position (0,4,18,8,20)
          and consonants on rest position of c
Step 11 : End
```

Function replace()

```
Step 1 : start
Step 2 : Print the message "Replace Vowel(*) and
         consonants(#)".
Step 3 : Run a for loop over string str from
         i=0 to length of string and repeat
         step 4 to step7 and do i++
Step 4 : Store character at ith index in
         variable ch.
Step 5 : If ch is ~~upper case~~ upper case do
         ch=ch-65
```

Scanned with CamScanner

Step 6: If ch is lower case do ch=ch-97.
Step 7: If ch is a vowel print (*) and if ch is consonant print (#) and if it is non alphabet print the charcter as it is.
Step 8: End

Function toggle()

Step 1: Start
Step 2: Print the message "Toggle" and change line
Step 3: Create a character array for word assuming length to be 200 and name it W.
Step 4: Call clear(W);
Step 5: declare an integer variable k & initialize it with zero.
Step 6: Start a for loop from i=0 to i less than length of str 7 and repeat step 9 to step and do i++.
Step 7: Store character at ith index in variable ch.
Step 8: If ch is an alphabet then W[k++] = toupper(ch)
Step 9: Else print first character of w in lower case and rest in uppercase and call clear(w) and put k=0.
Step 10: End

Scanned with CamScanner

Function line()

Step 1: Start
Step 2: Print the message "Printing the words, one in a line"
Step 3. Define a variable flag and initialize it with 0.
Step 4: Start a for loop from i=0 to i less than length of str and repeat step to step and do i=i+1.
Step 5: Store character at ith index of str in variable ch.
Step 6: If ch is an alpha numeral print it and flag=0.
Step 7: Else if flag=0 print ch and change line and do flag = 1.
Step 8: End

Function search().

Step 1: Start
Step 2: Print the message "Search"
Step 3: Input the character to be searched in a variable key.
Step 4: Create a char array w of 200 size and call ~~dar~~ clear(w) and. do k=0.
Step 5: Initialize fear variable flag=0 & flag2=0.
Step 6: Print message "Output".
Step 7: Execute a for loop from i=0 to i<strlen(str) and repeat step 8 to step 12 and do i=i+1.

Scanned with CamScanner

```
Step 8 : Store ith character of str in a variable
         ch
Step 9 : If ch is alphabet do w[k++] = ch
         and if (ch == key) do flag = 1.
Step 10 : Else, if (flag is 1) go and check flag2 if
          it is 0 print w else print w along with ','.
Step 11 : call clear(w) and put k=0;
Step 12 : End.
```

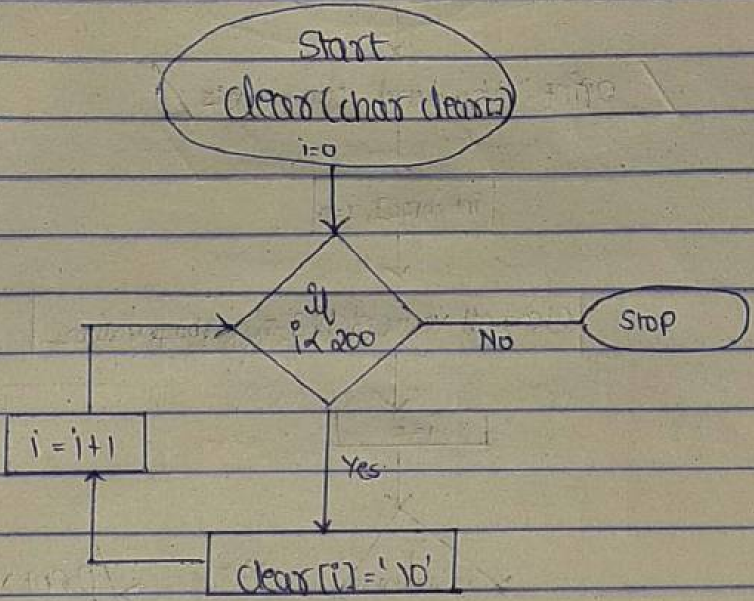
Function main()

```
Step 1 : start
Step 2 : Enter the string in variable str from user
         and limit it to 1000 size.
Step 3 : call count char();
Step 4 : call toggle();
Step 5 : call vandc();
Step 6 : call search();
Step 7 : call replace();
Step 8 : call line();
Step 9 : End.
```

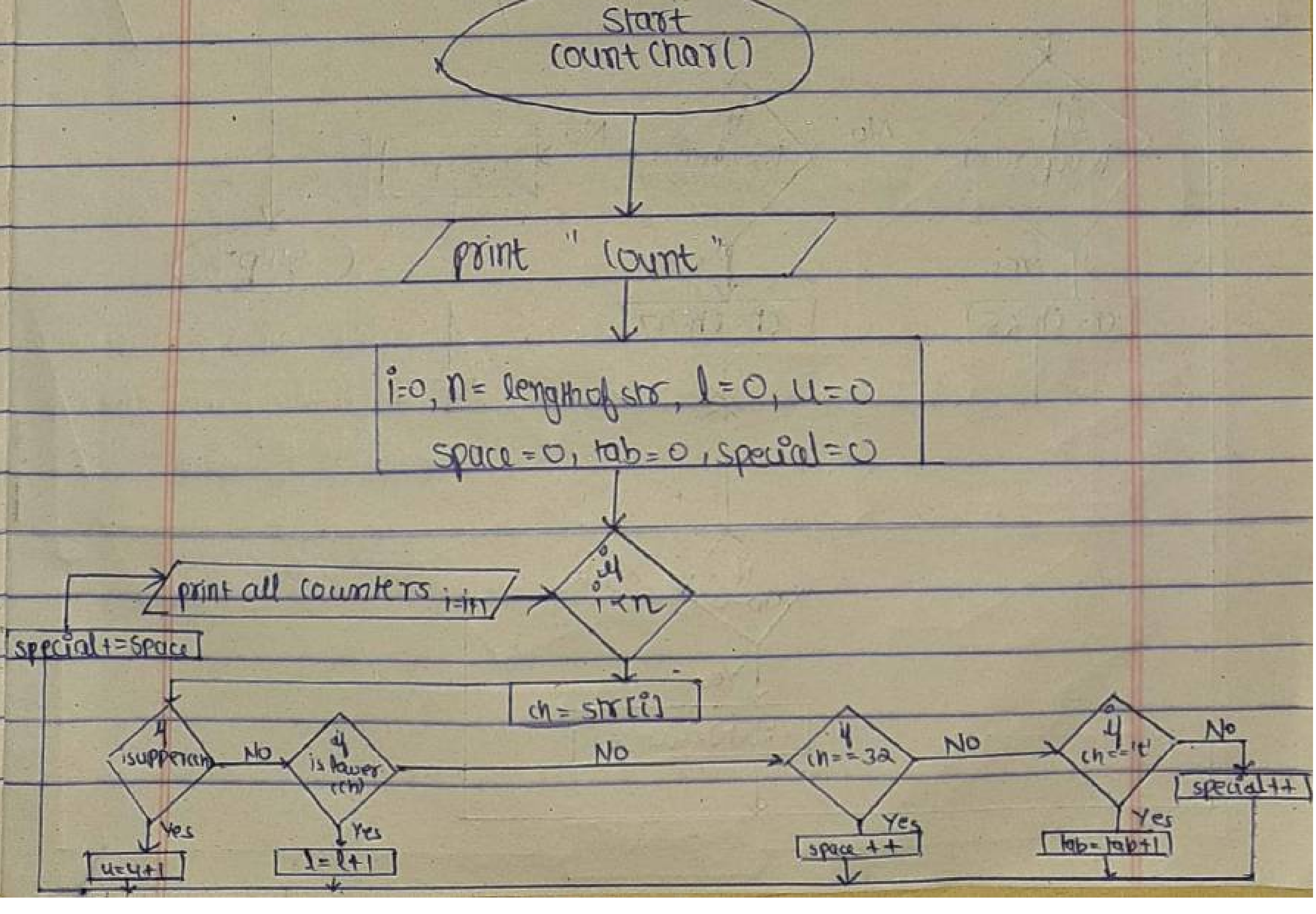
Flowchart

char Str[1000]
↳ global variable

clear function

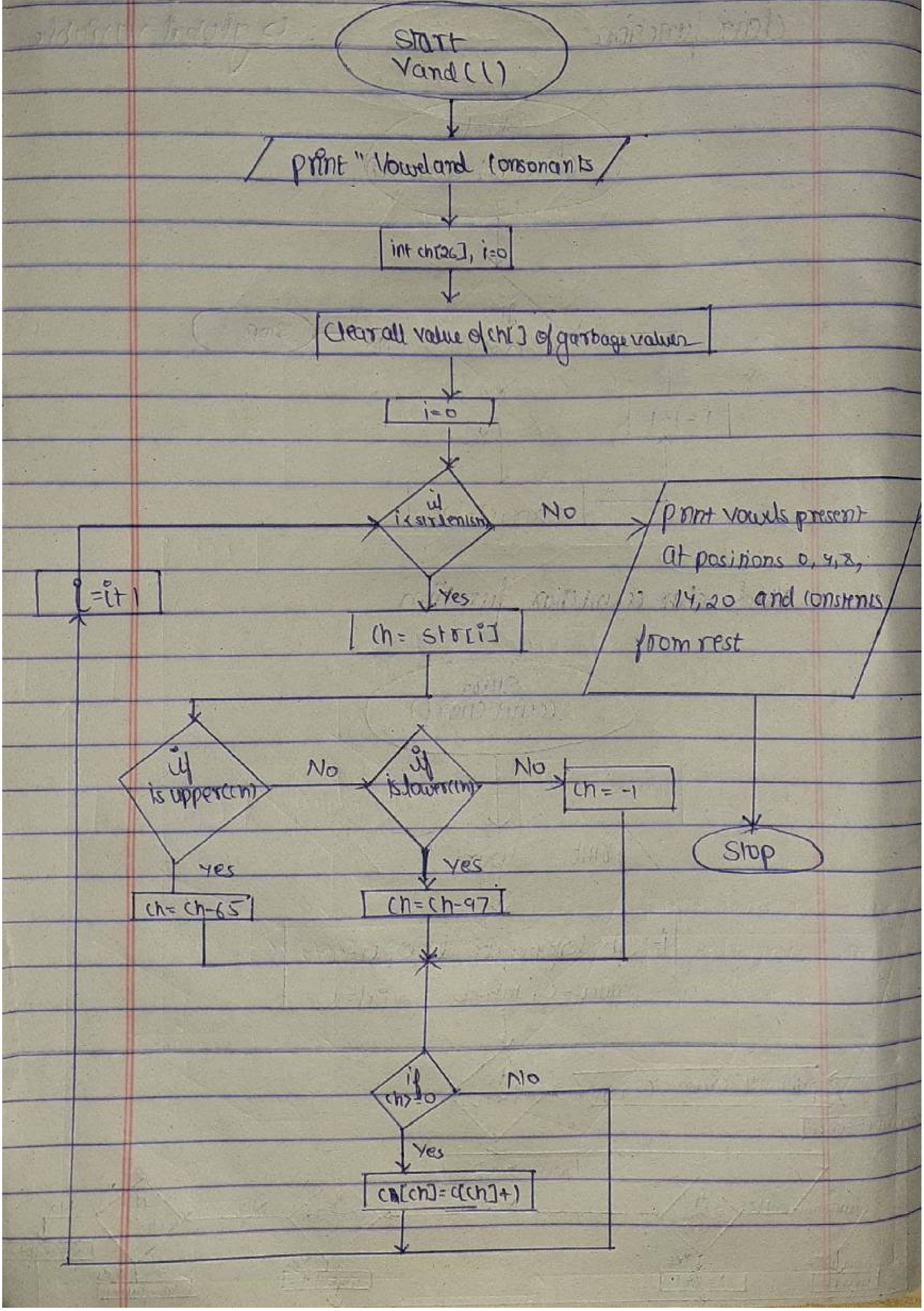


~~void~~ ~~char~~ countchar function



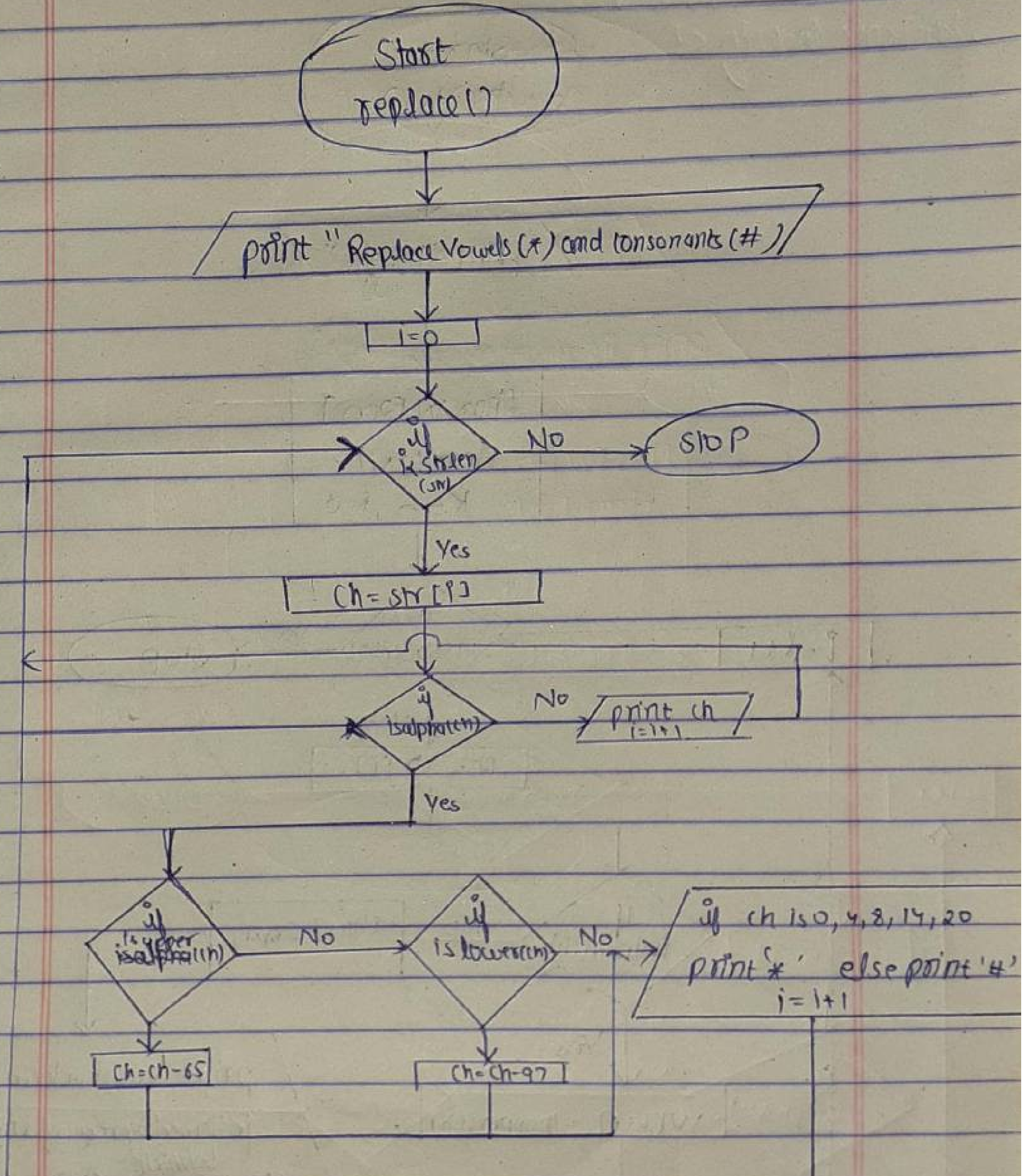
Scanned with CamScanner

Vand c function



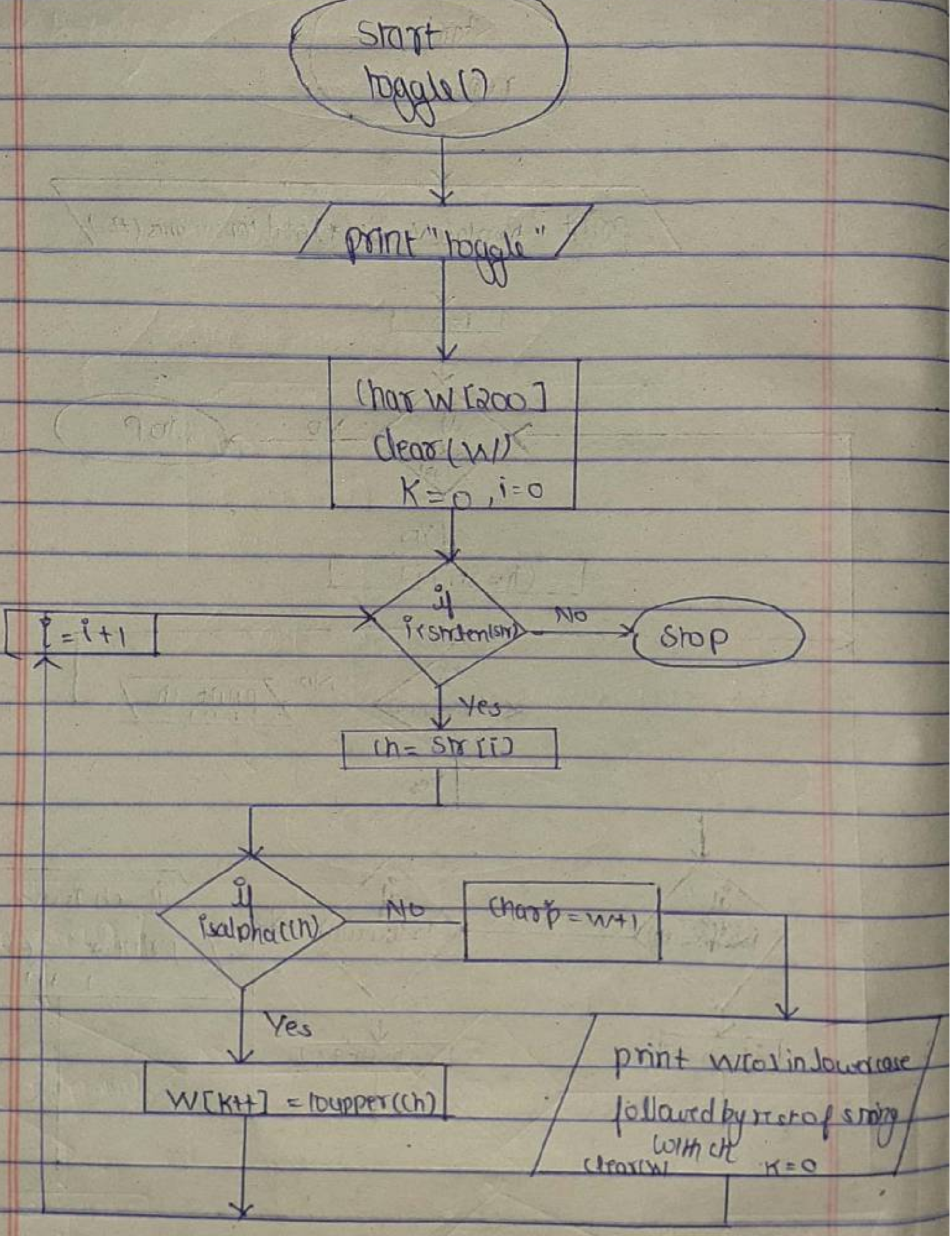
Scanned with CamScanner

replace function

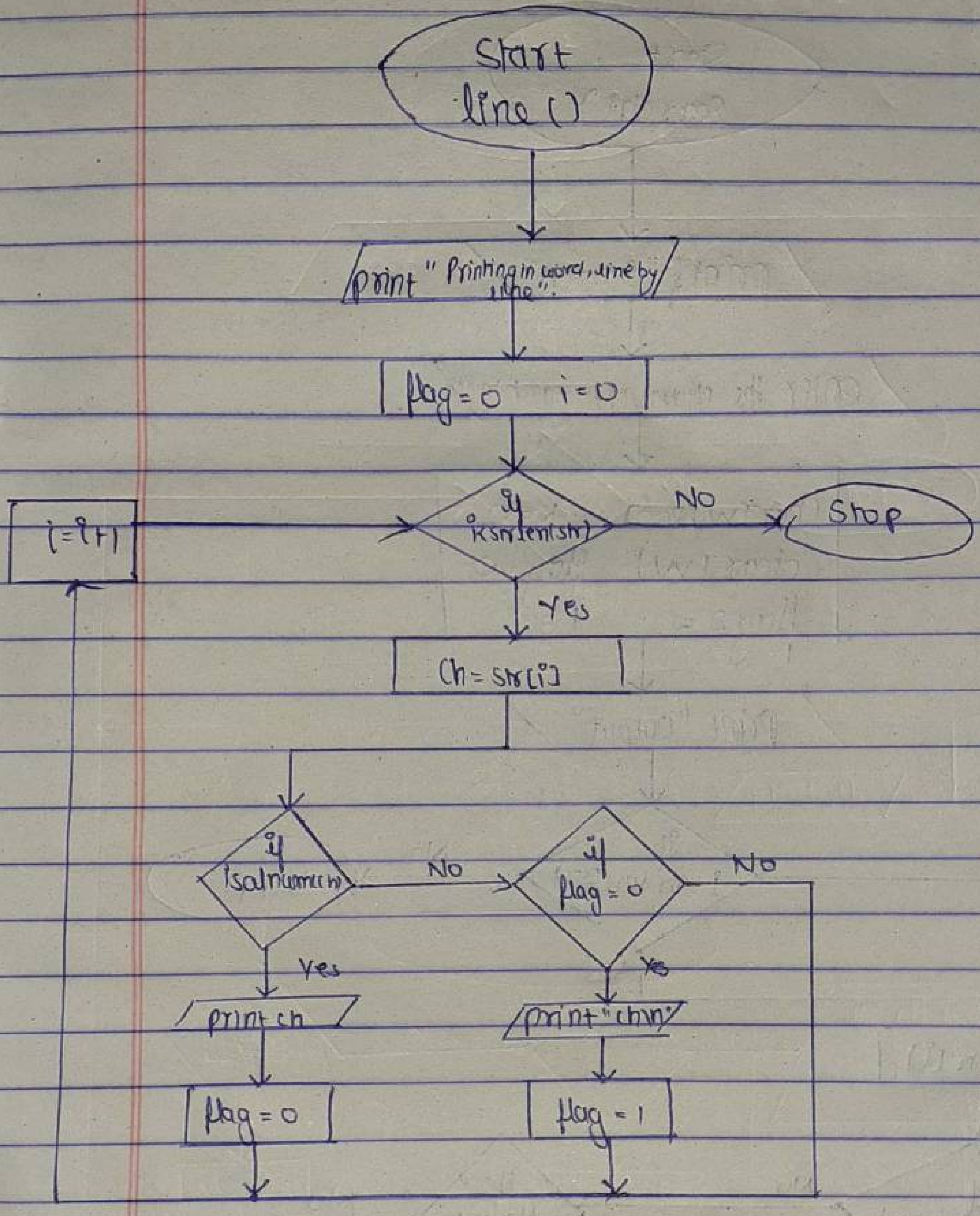


Scanned with CamScanner

toggle function

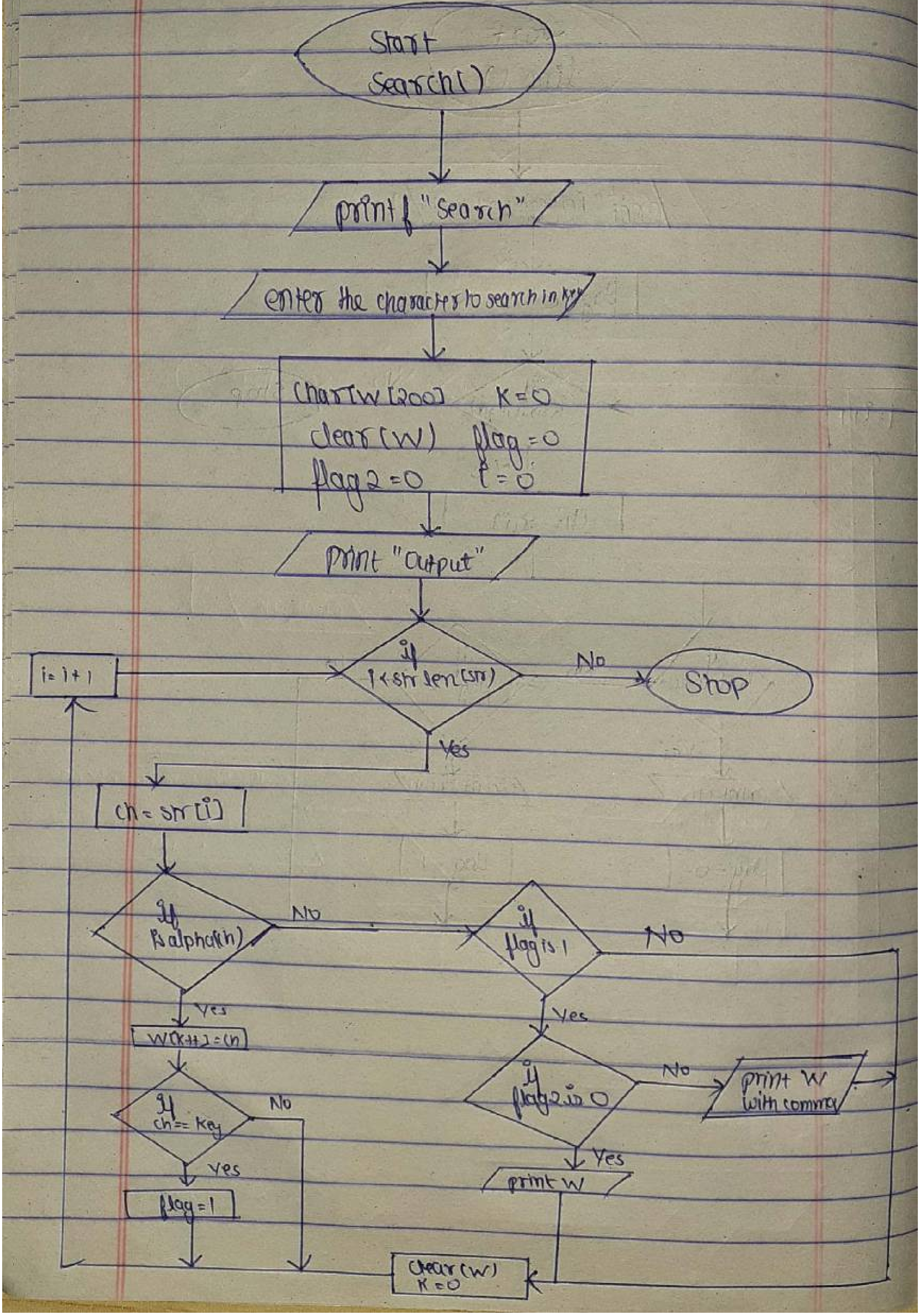


void line



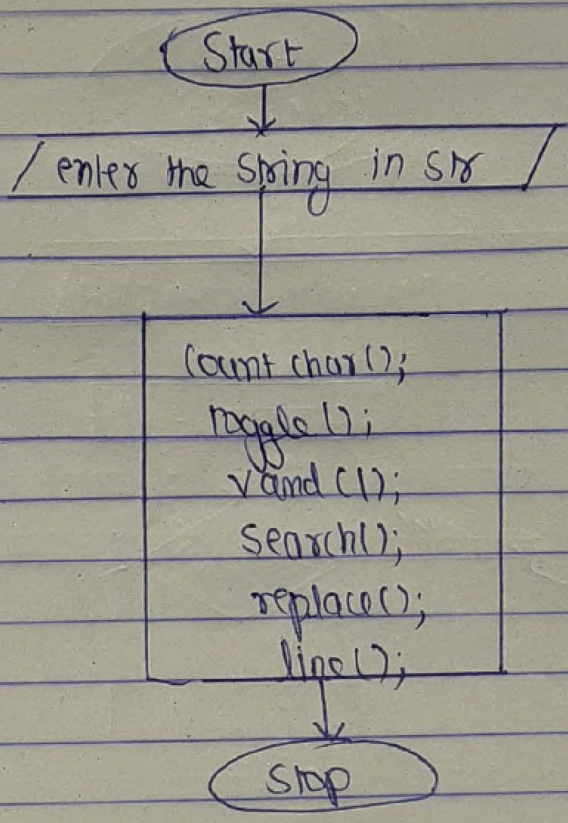
lowercase
f string

search function



Scanned with CamScanner

main function



Scanned with CamScanner