**SECURE CODING**

**ASSIGNMENT 11 – FINDING VULNERABILITIES IN THE GIVEN CODE**

**NAME : SREENIDHI GANACHARI**

**REGISTRATION NUMBER : 19BCE7230**

**SLOT : L – 39 + 40**

**1. Find out the vulnerabilities of the code and fix it.**

#include <stdio.h>

int main () {

char user\_name[10];

int allow = 0;

printf("Enter your username, please: ");

gets(user\_name);

if (grantAccess(user\_name)) {

allow = 1;

}

if (allow != 0) {

printf("%s",user);

// privilegedAction();

}

return 0;

}

int grantAccess(char \*name)

{

char name1[]="vitap";

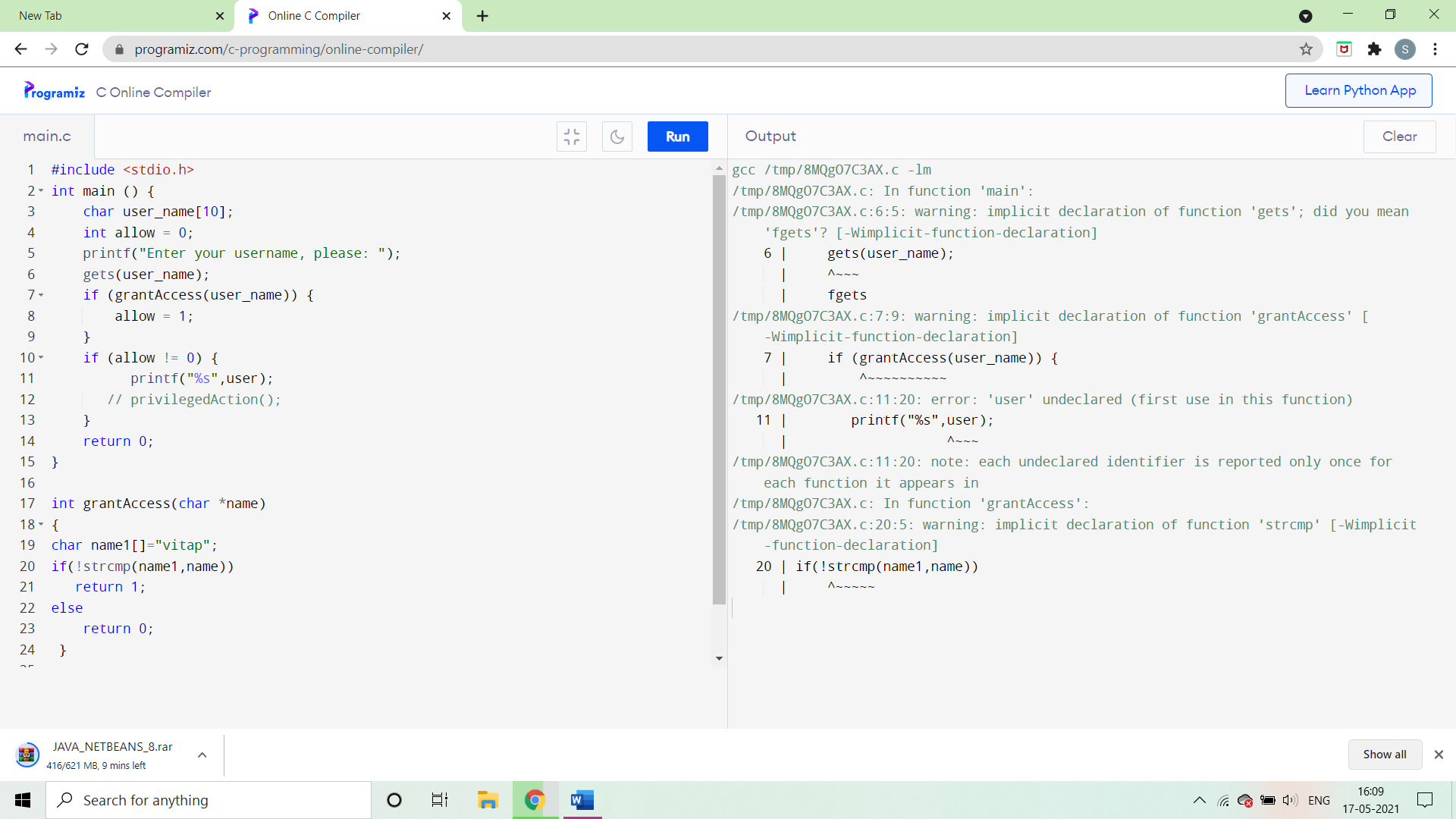
if(!strcmp(name1,name))

return 1;

else

return 0;

}

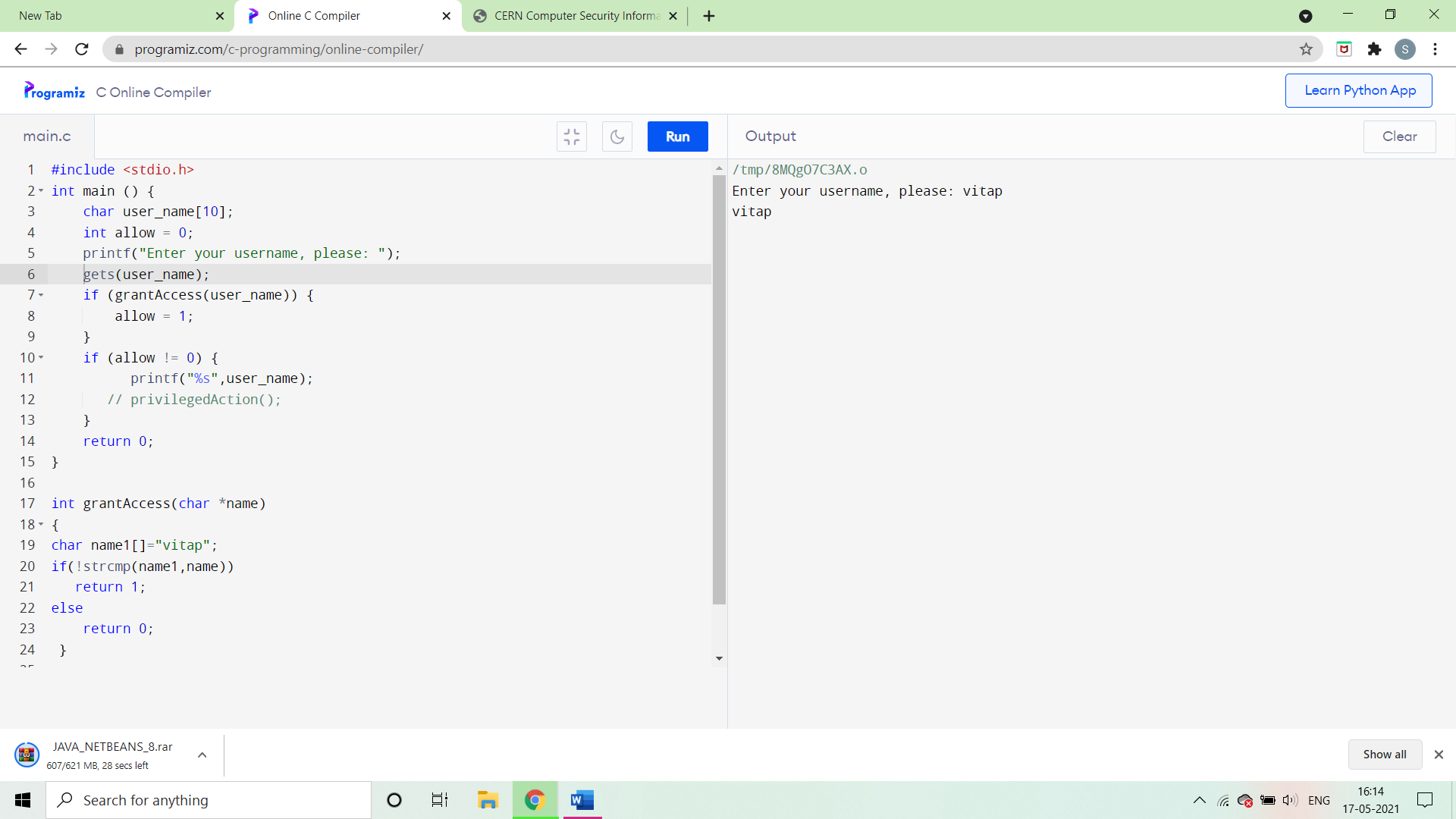


The problem in the above program is that stdio () function doesnot check for the length of the buffer which leads to the vulnerability .

**The correct code would be –**

We can change gets to fgets and the word user (11) can be written as user\_name because user is not defined initially .

Once we fix this , we enter the username as vitap and then it grants us access to proceed further .



**2. Find out the vulnerabilities of the code and fix it.**

#include <stdio.h>

#include <stdlib.h>

enum { BUFFER\_SIZE = 10 };

int main() {

char buffer[BUFFER\_SIZE];

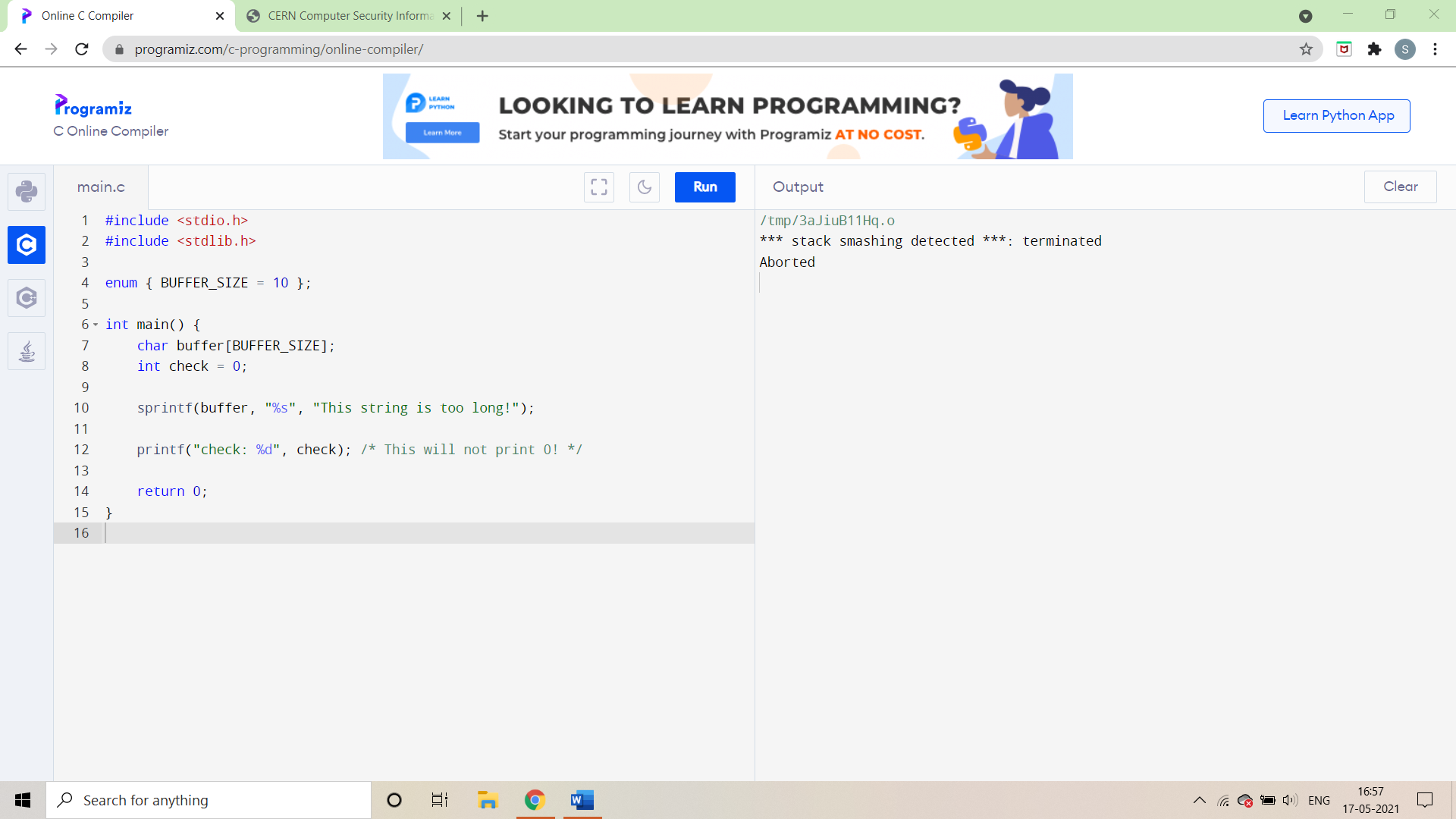
int check = 0;

sprintf(buffer, "%s", "This string is too long!");

printf("check: %d", check); /\* This will not print 0! \*/

return 0;

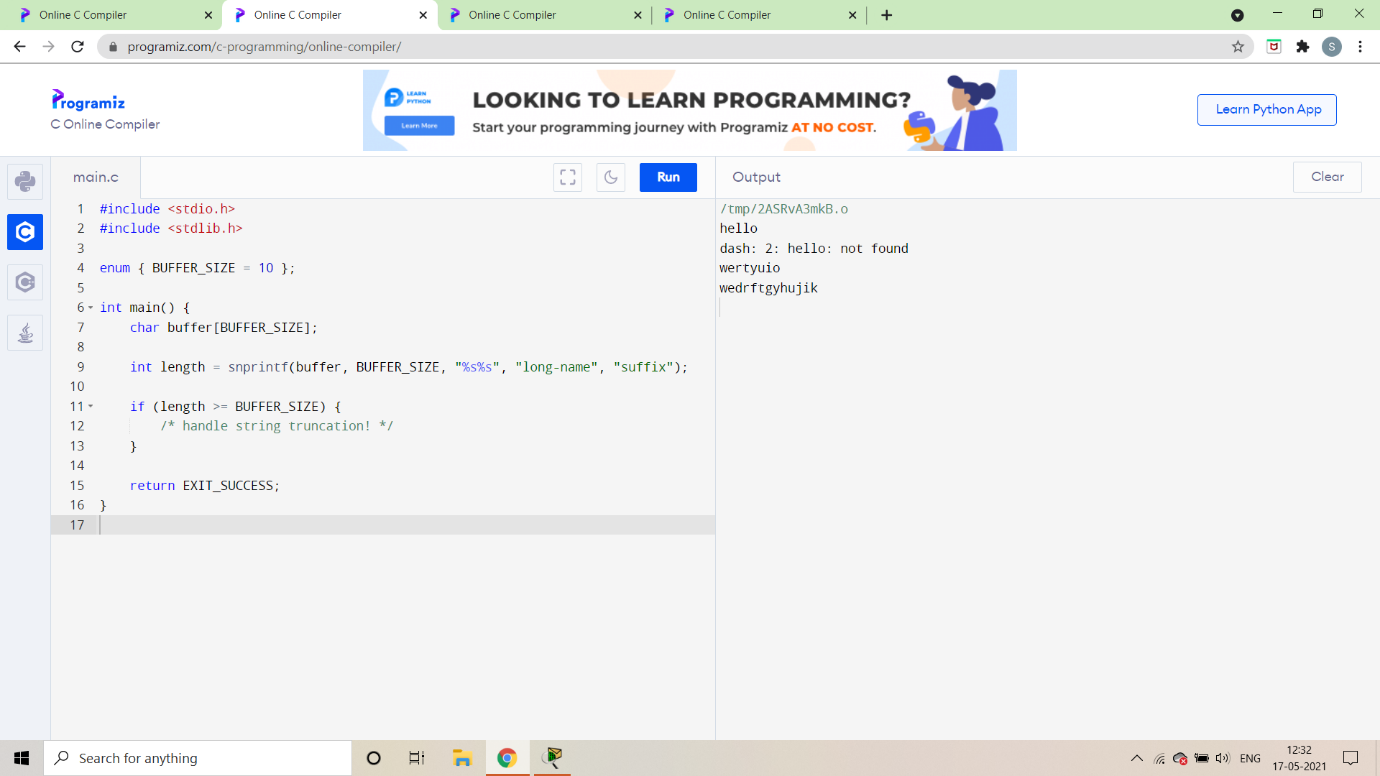
}



In the above given code it causes buffer overflow as the sprint command does not check the buffer boundaries and it thus it causes the overflow vulnerability .

**The correct code would be –**

The sprintf is changed to snprintf because it prevents buffers overflows and returning the minimal size of buffer needed to fit the whole formatted string.



**3. Find out the vulnerabilities of the code and fix it.**

#include <stdio.h>

int main () {

sort int sub\_marks[10]; //each subject marks is less than 10

sort int sum=0;

int i;

printf("Enter the 10 subjects marks");

for(i=0;i<10;i++)

scanf("%d",&roll\_numbers[i]);

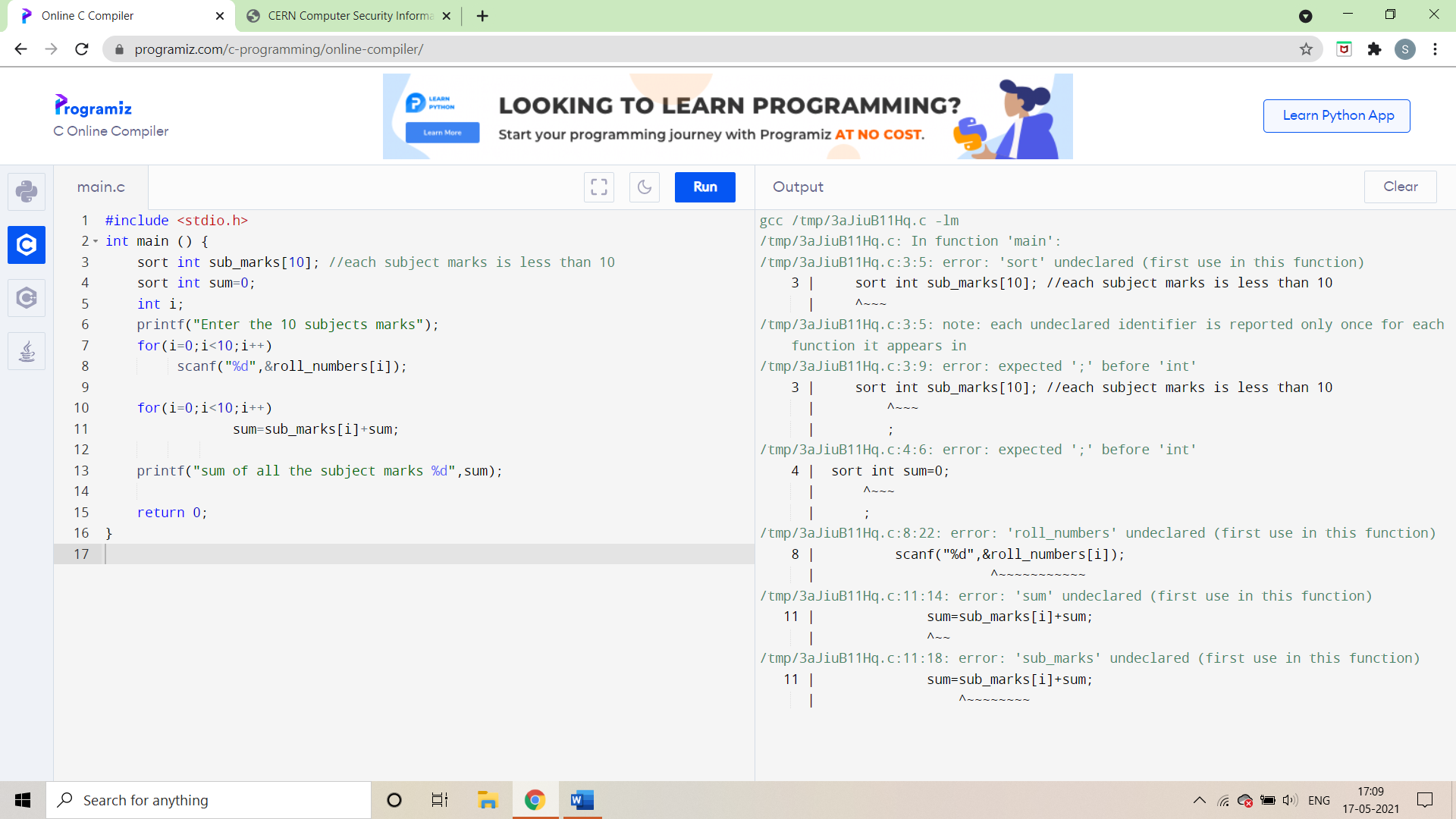
for(i=0;i<10;i++)

sum=sub\_marks[i]+sum;

printf("sum of all the subject marks %d",sum);

return 0;

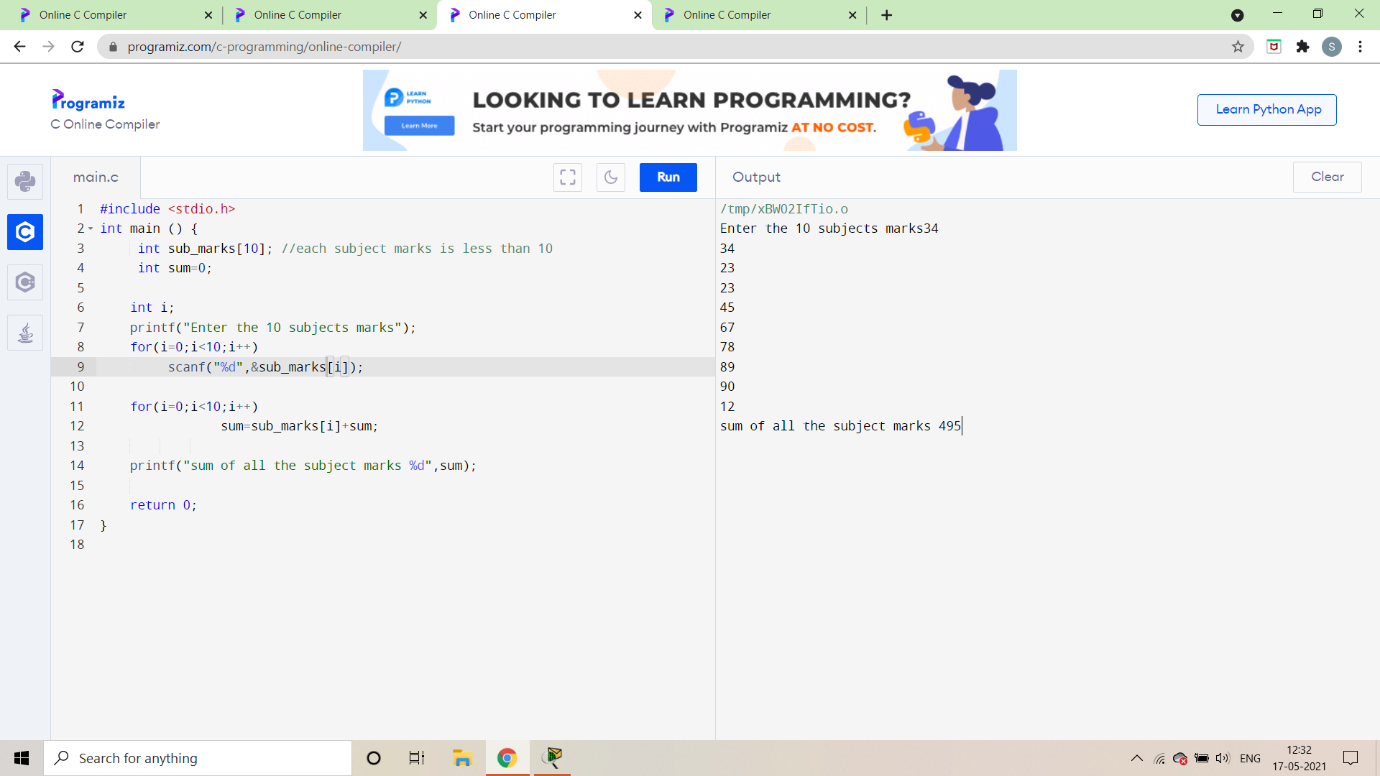
}



The function sort is not defined in the above code , also the term roll\_numbers is not initialized .

**The correct code would be –**

The (9) line is corrected in the above code which enables the entries .



**4. Find out the vulnerabilities of the code and fix it.**

#include <stdio.h>

int main (int argc, char \*argv[]) {

char name1[10]; //each subject marks is less than 10

char name2[10];

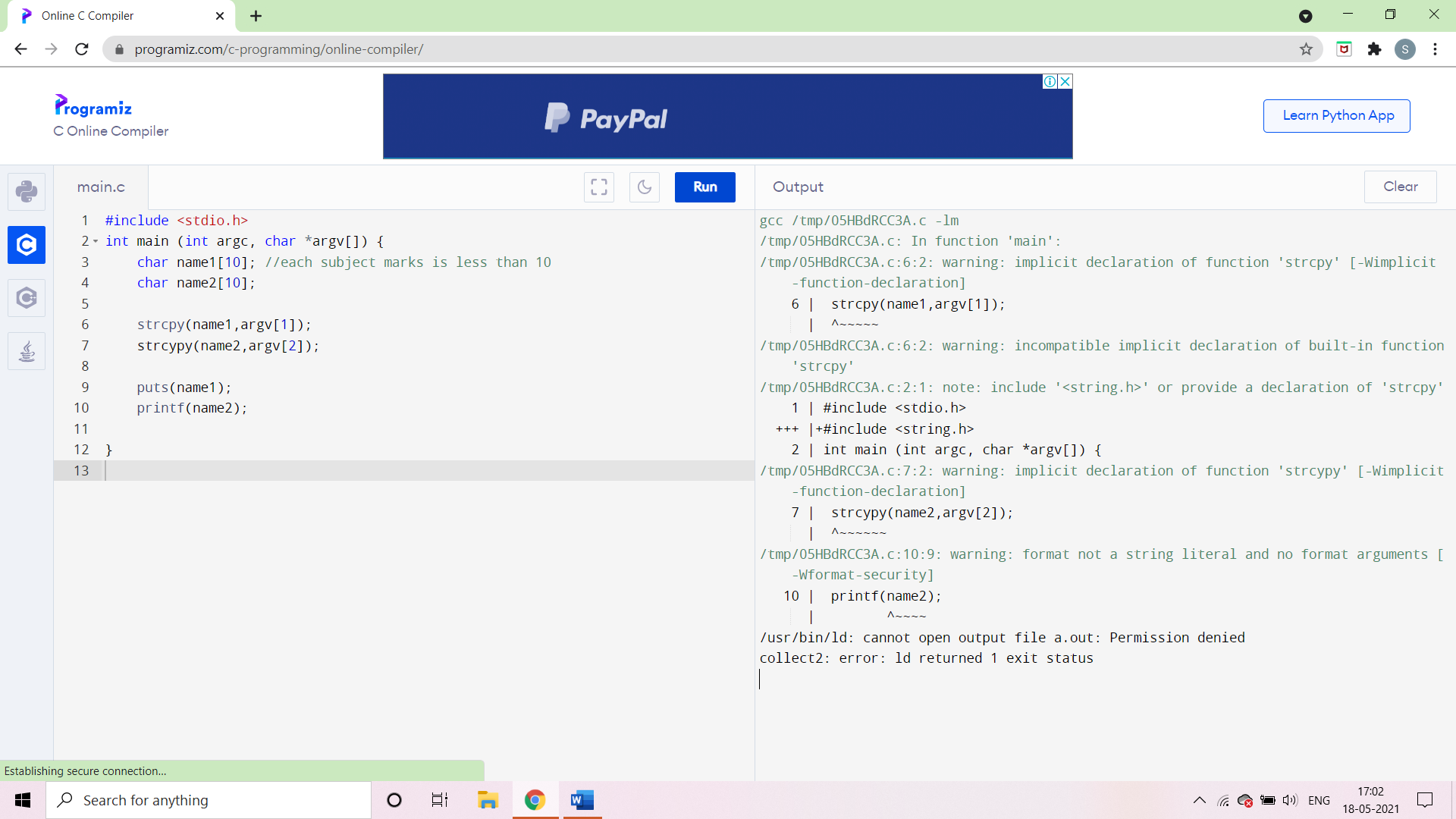
strcpy(name1,argv[1]);

strcypy(name2,argv[2]);

puts(name1);

printf(name2);

}



The string in the above code has the wrong format , and also implicit declaration of the functions strcypy .

**The correct code would be –**

In the above code we declare functions for the strcpy and can thus give the input successfully.

