Batch:C1-1	**Roll No.:16010123012**
Experiment / assignment / tutorial No. 5	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE: Write a program in C to demonstrate use of character arrays and strings

AIM:

a) Write a program that searches for a substring within a given string.

b) Write a program to check if one string is the rotation of another.

Expected OUTCOME of Experiment:

Apply the concepts of arrays and strings(CO3).

Books/ Journals/ Websites referred:

1. Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.
2. Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill.
3. Introduction to programming and problem solving , G. Michael Schneider ,Wiley India edition.

Problem Definition:

1. The program searches for a substring within a given string and returns the starting index if found, or -1 otherwise.

Example:

Test case 1: Input: String: Programming Substring: ing Output: Index: 8	Test case 2: Input: String: Programming Substring: Python Output: Index: -1

2. The program checks whether a given string is the rotation of the other.

Example:

Test case 1: Input: String 1: abcd String 2: bcda Output: Yes	Test case 2: Input: String 1: abcd Substring: dcba Output: No

Algorithm:

1.

1. Include the necessary header file 'stdio.h'.
2. Define the main function.
3. Declare character arrays s1, s2, and s3 each of size 10.
4. Print "Enter string:".
5. Read a string from the user into the array s2 using scanf.
6. Check if the length of the input string (s2) is not equal to 5.
   a. If true, print "NO" and exit the program.
   b. If false, continue to the next step.
7. Concatenate the strings in arrays s1 and s2 into a new string s5 using strcat.
8. Declare a new character pointer s4 and use strstr to find the occurrence of the substring s3 in the concatenated string s5.
9. Check if the result pointer s4 is not equal to 0.
   a. If true, print "YES" as the substring is found in the concatenated string.
   b. If false, print "NO" as the substring is not found in the concatenated string.

2.

1. Include the necessary header file ‘stdio.h’.
2. Define the main function.
3. Declare character arrays s1, s2, and s3 each of size 10.
4. Print "Enter string:".
5. Read a string from the user into the array s2 using scanf.
6. Check if the length of the input string (s2) is not equal to 5.
   a. If true, print "NO" and exit the program.
   b. If false, continue to the next step.
7. Concatenate the strings in arrays s1 and s2 into a new string s5 using strcat.
8. Declare a new character pointer s4 and use strstr to find the occurrence of the substring s3 in the concatenated string s5.
9. Check if the result pointer s4 is not equal to 0.
   a. If true, print "YES" as the substring is found in the concatenated string.
   b. If false, print "NO" as the substring is not found in the concatenated string.

Implementation details:

1.

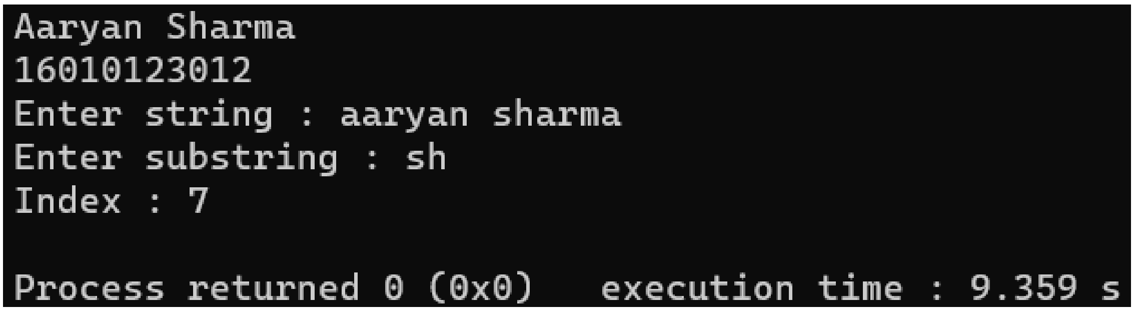
```
#include<stdio.h>
int main()
{
   printf("Aaryan Sharma\n");
   printf("16010123012\n");

   char str1[100];
   char str2[100];
   printf("Enter string : ");
   gets(str1);
   printf("Enter substring : ");
   scanf("%s", &str2);

   char *r=strstr(str1,str2);
   if(r){
      printf("Index : %d\n",r-str1);
   }else{
   printf("-1");
   }
   return 0;
}
```

2.

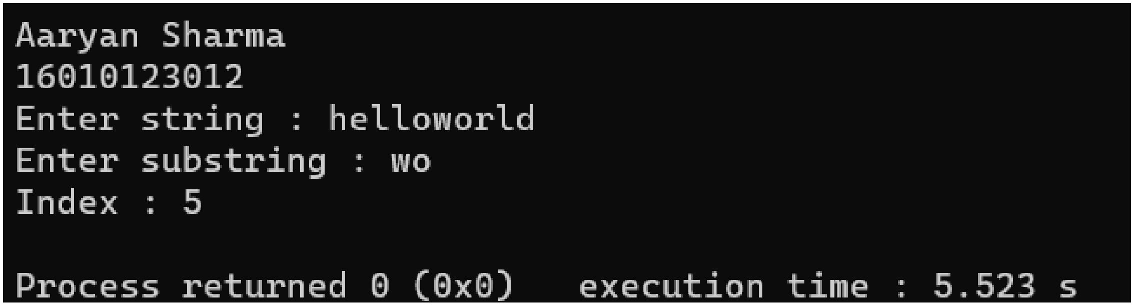
```
#include<stdio.h>
int main()
{
   printf("Aaryan Sharma\n");
   printf("16010123012\n");

   char s1[10]="abcde";
   char s2[10]="abcde";
   char s3[10];
   printf("Enter string: ");
   scanf("%s",&s2);
   if(strlen(s2)!=5){
      printf("NO");
   }else{
      char *s5=strcat(s1,s2);
      char *s4=strstr(s5,s3);
   if(s4!=0){
      printf("YES");
```

```
    }else{
        printf("NO");
    }
    return 0;
}
}
```

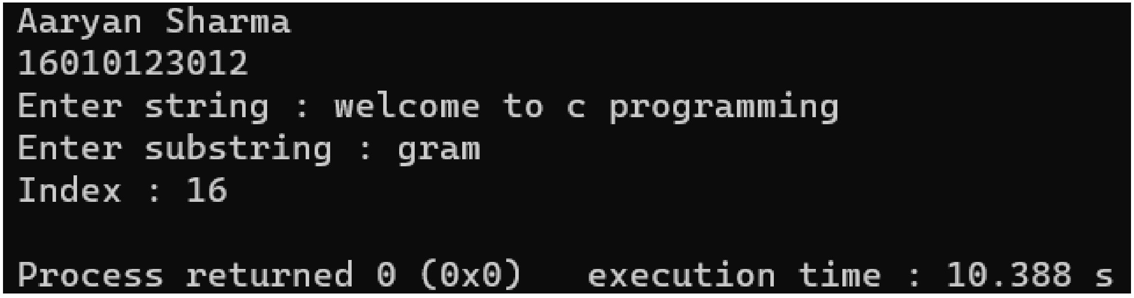
Output(s):
1.



```
Aaryan Sharma
16010123012
Enter string : aaryan sharma
Enter substring : sh
Index : 7

Process returned 0 (0x0)   execution time : 9.359 s
```

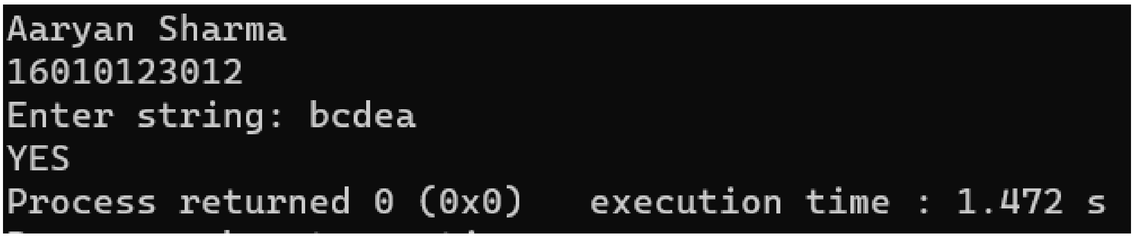


```
Aaryan Sharma
16010123012
Enter string : helloworld
Enter substring : wo
Index : 5

Process returned 0 (0x0)   execution time : 5.523 s
```

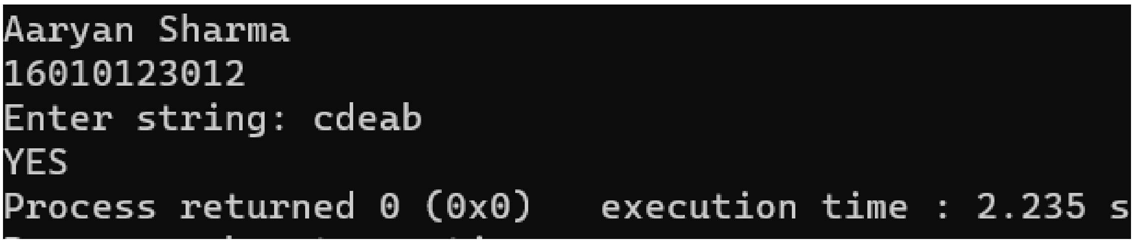


```
Aaryan Sharma
16010123012
Enter string : welcome to c programming
Enter substring : gram
Index : 16

Process returned 0 (0x0)   execution time : 10.388 s
```

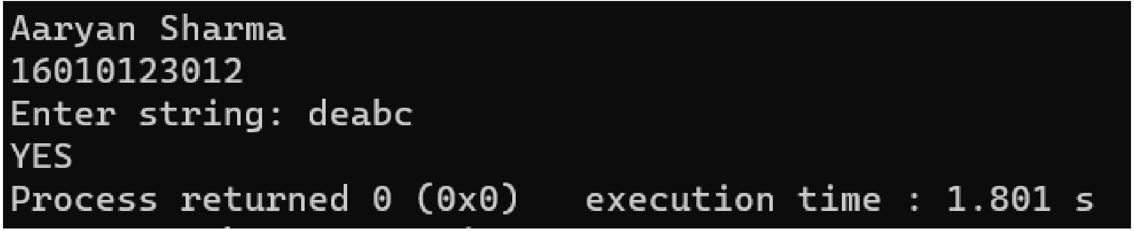
2.



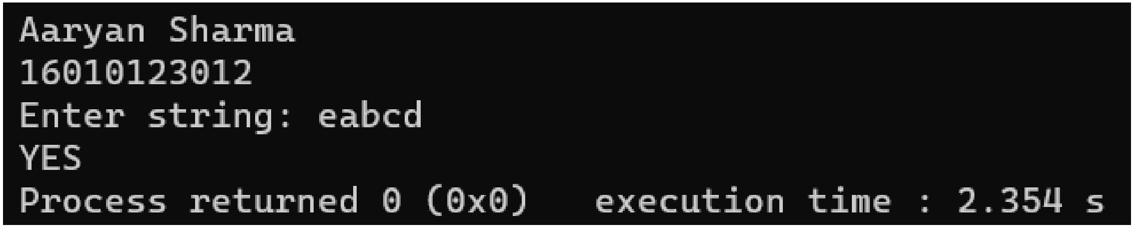
```
Aaryan Sharma
16010123012
Enter string: bcdea
YES
Process returned 0 (0x0)   execution time : 1.472 s
```



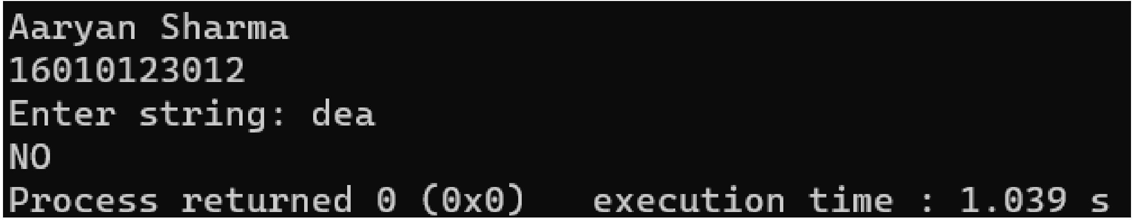
```
Aaryan Sharma
16010123012
Enter string: cdeab
YES
Process returned 0 (0x0)   execution time : 2.235 s
```



```
Aaryan Sharma
16010123012
Enter string: deabc
YES
Process returned 0 (0x0)   execution time : 1.801 s
```



```
Aaryan Sharma
16010123012
Enter string: eabcd
YES
Process returned 0 (0x0)   execution time : 2.354 s
```



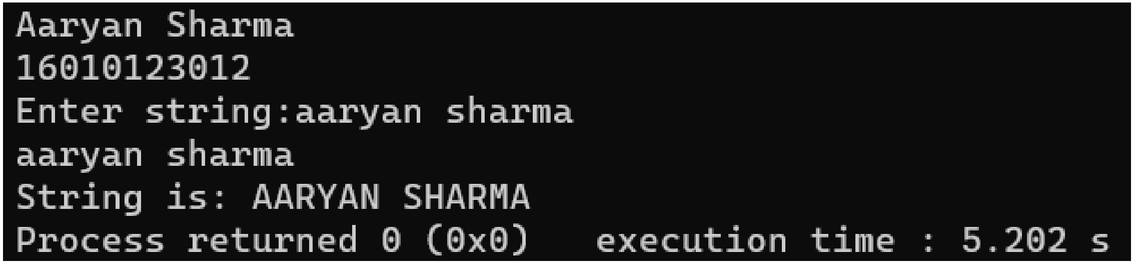
```
Aaryan Sharma
16010123012
Enter string: dea
NO
Process returned 0 (0x0)   execution time : 1.039 s
```

Conclusion:
We have successfully performed the experiment to demonstrate use of character arrays and strings in C programming. Through the development of various programs, we explored the fundamental operations associated with character arrays and strings, including input/output, concatenation, and searching.

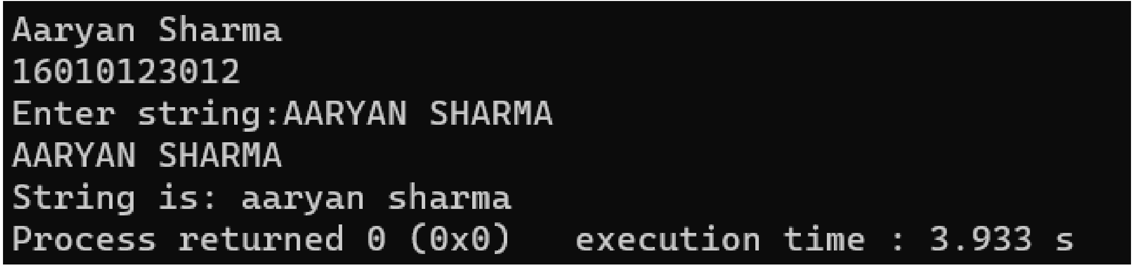
Post Lab Questions

1. Write a C program to toggle case of each character in a string i.e. if a character is in uppercase, change it to lower case and vice-versa.

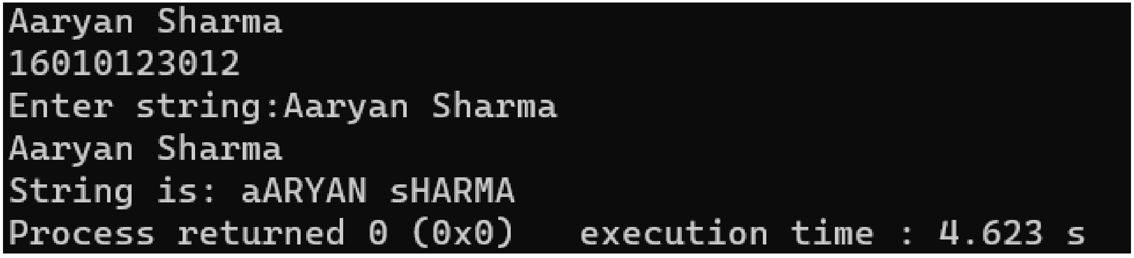
```
#include<stdio.h>
int main(){
printf("Aaryan Sharma\n");
printf("16010123012\n");
char str[1000];
int i;
printf("Enter string:");
gets(str);
printf("%s",str);
for(i=0;str[i];i++){
   if(str[i]>=97 && str[i]<=122)
     {
          str[i]-=32;
}
else if(str[i]>=65 && str[i]<=90){
str[i]+=32;
}
}
printf("\nString is: %s",str);
return 0;
}
```



```
Aaryan Sharma
16010123012
Enter string:aaryan sharma
aaryan sharma
String is: AARYAN SHARMA
Process returned 0 (0x0)   execution time : 5.202 s
```

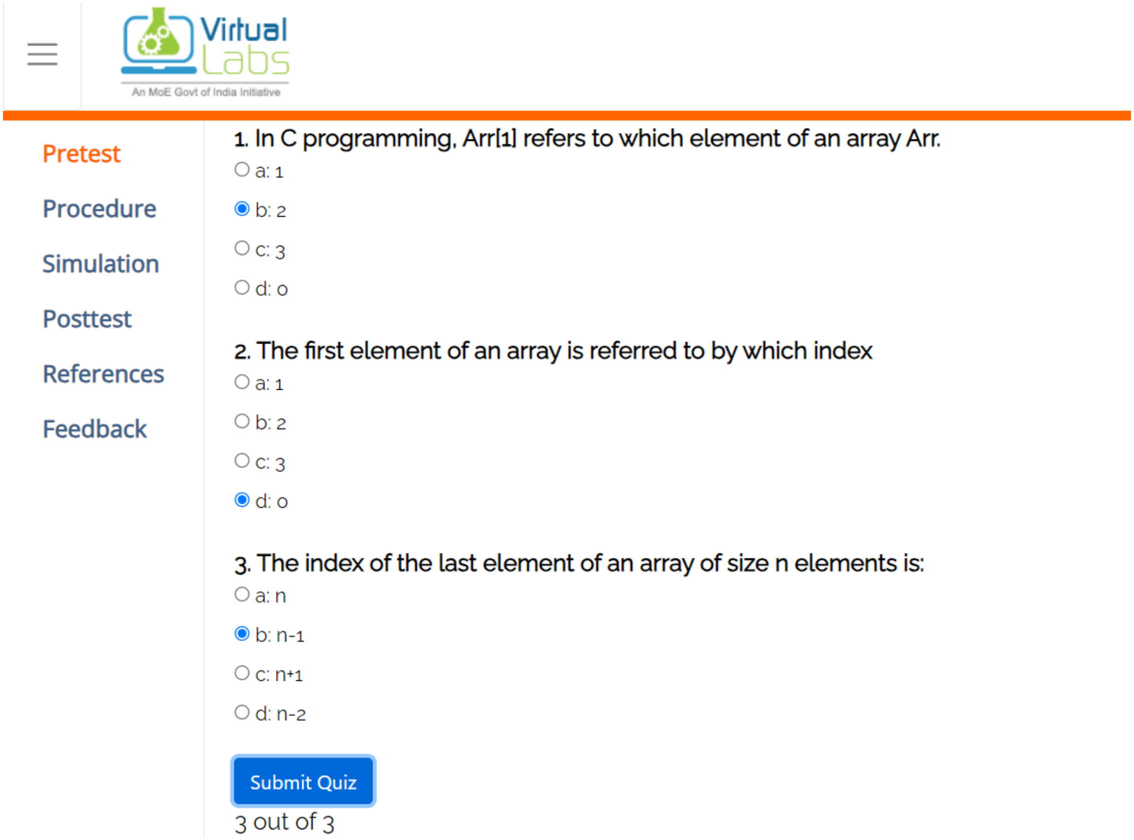


```
Aaryan Sharma
16010123012
Enter string:AARYAN SHARMA
AARYAN SHARMA
String is: aaryan sharma
Process returned 0 (0x0)   execution time : 3.933 s
```



```
Aaryan Sharma
16010123012
Enter string:Aaryan Sharma
Aaryan Sharma
String is: aARYAN sHARMA
Process returned 0 (0x0)   execution time : 4.623 s
```

2. Virtual Lab for Arrays
https://cse02-iiith.vlabs.ac.in/exp/arrays/simulation.html



1. In C programming, Arr[1] refers to which element of an array Arr.

○ a: 1

● b: 2

○ c: 3

○ d: 0

2. The first element of an array is referred to by which index

○ a: 1

○ b: 2

○ c: 3

● d: 0

3. The index of the last element of an array of size n elements is:

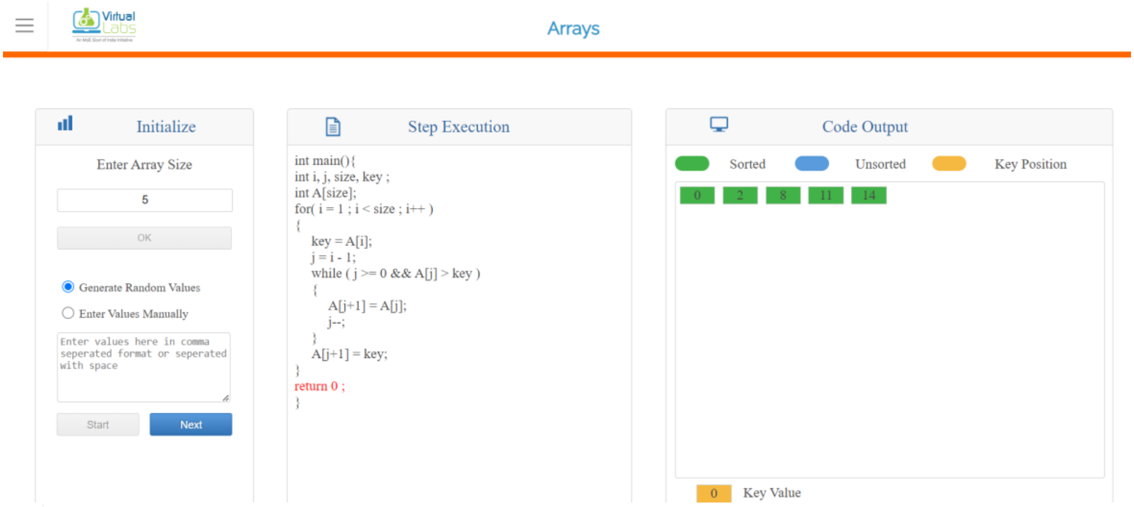
○ a: n

● b: n-1

○ c: n+1

○ d: n-2

3 out of 3



Enter Array Size: 5

● Generate Random Values

○ Enter Values Manually

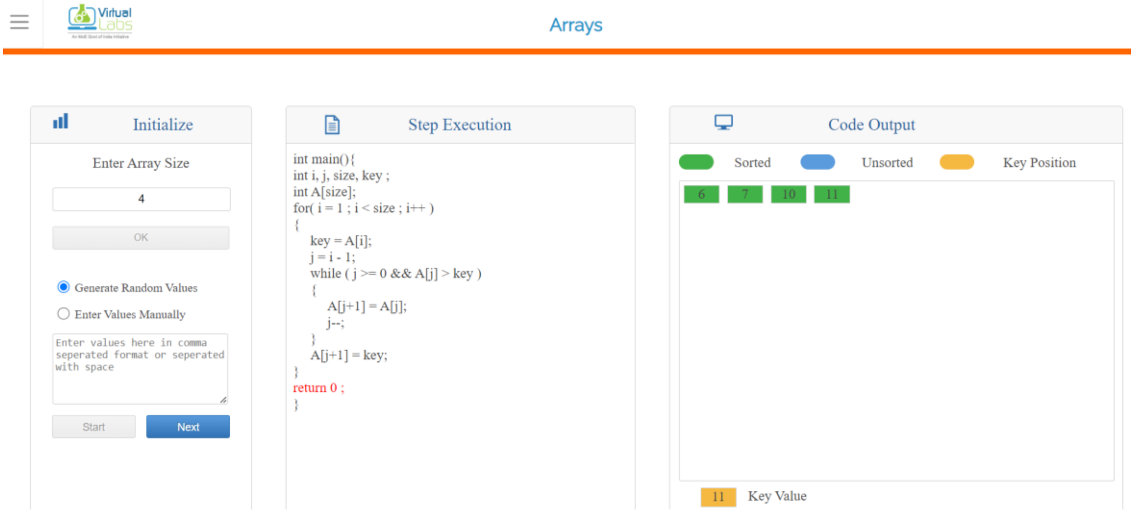
Enter values here in comma seperated format or seperated with space

```
int main(){
int i, j, size, key ;
int A[size];
for( i = 1 ; i < size ; i++ )
{
    key = A[i];
    j = i - 1;
    while ( j >= 0 && A[j] > key )
    {
        A[j+1] = A[j];
        j--;
    }
    A[j+1] = key;
}
return 0 ;
}
```

🟩 Sorted 🟦 Unsorted 🟨 Key Position

0	2	8	11	14

0 Key Value



Enter Array Size: 4

● Generate Random Values

○ Enter Values Manually

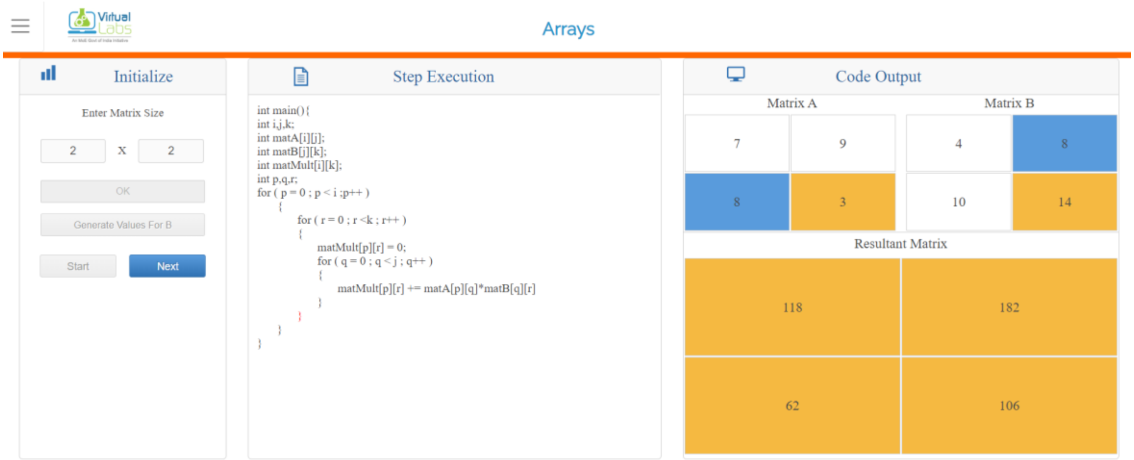
Enter values here in comma seperated format or seperated with space

```
int main(){
int i, j, size, key ;
int A[size];
for( i = 1 ; i < size ; i++ )
{
    key = A[i];
    j = i - 1;
    while ( j >= 0 && A[j] > key )
    {
        A[j+1] = A[j];
        j--;
    }
    A[j+1] = key;
}
return 0 ;
}
```

🟩 Sorted 🟦 Unsorted 🟨 Key Position

6	7	10	11

11 Key Value



```
int main(){
int i,j,k;
int matA[i][j];
int matB[j][k];
int matMult[i][k];
int p,q,r;
for ( p = 0 ; p < i ;p++ )
    {
        for ( r = 0 ; r <k ; r++ )
        {
            matMult[p][r] = 0;
            for ( q = 0 ; q < j ; q++ )
            {
                matMult[p][r] += matA[p][q]*matB[q][r]
            }
        }
    }
}
```

Matrix A

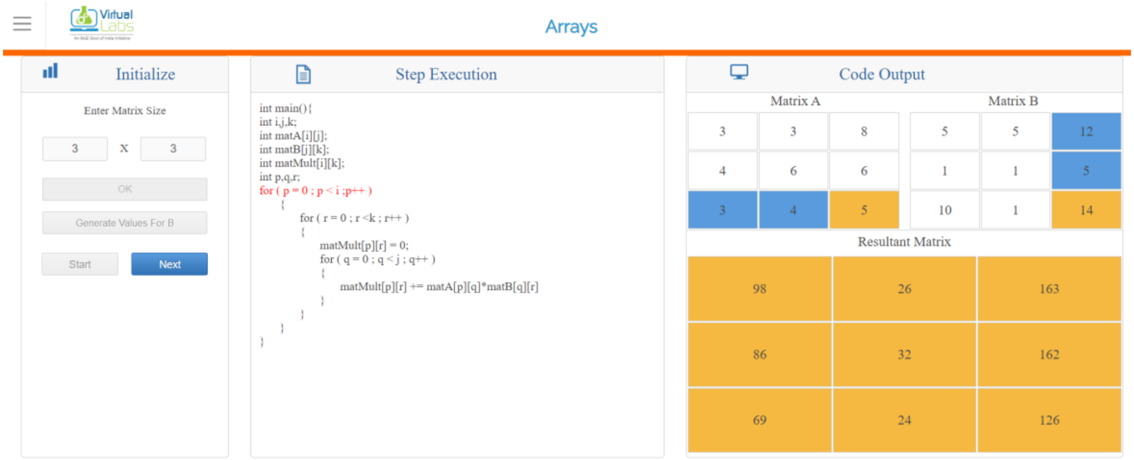
7	9
8	3

Matrix B

4	8
10	14

Resultant Matrix

118	182
62	106



```
int main(){
int i,j,k;
int matA[i][j];
int matB[j][k];
int matMult[i][k];
int p,q,r;
for ( p = 0 ; p < i ;p++ )
    {
        for ( r = 0 ; r <k ; r++ )
        {
            matMult[p][r] = 0;
            for ( q = 0 ; q < j ; q++ )
            {
                matMult[p][r] += matA[p][q]*matB[q][r]
            }
        }
    }
}
```

Matrix A

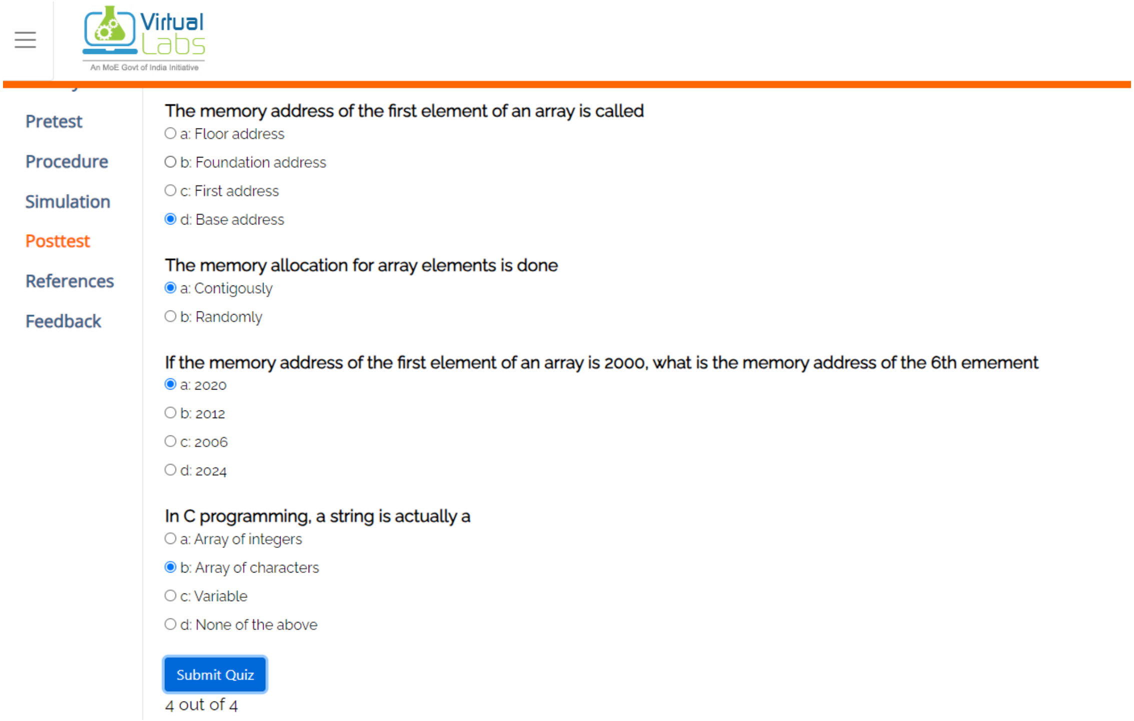
3	3	8
4	6	6
3	4	5

Matrix B

5	5	12
1	1	5
10	1	14

Resultant Matrix

98	26	163
86	32	162
69	24	126



The memory address of the first element of an array is called

○ a: Floor address
○ b: Foundation address
○ c: First address
● d: Base address

The memory allocation for array elements is done

● a: Contigously
○ b: Randomly

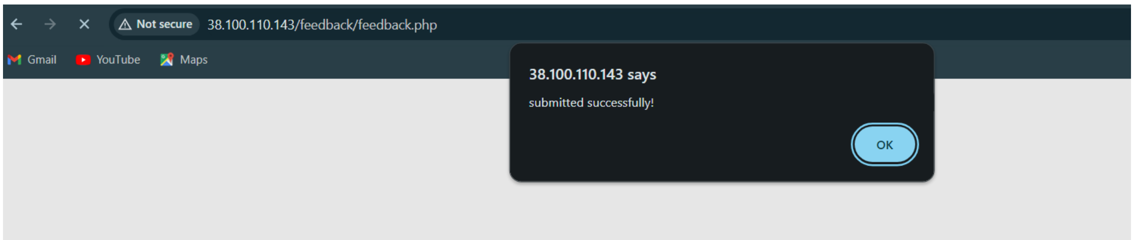
If the memory address of the first element of an array is 2000, what is the memory address of the 6th emement

● a: 2020
○ b: 2012
○ c: 2006
○ d: 2024

In C programming, a string is actually a

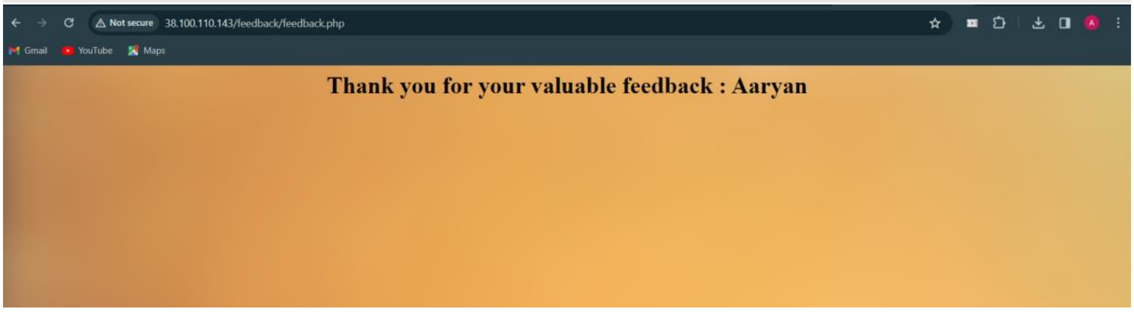
○ a: Array of integers
● b: Array of characters
○ c: Variable
○ d: None of the above

4 out of 4



38.100.110.143 says

submitted successfully!



Thank you for your valuable feedback : Aaryan

Date: 18/02/2024 **Signature of faculty in-charge**