

Programming in C

Strings

DPP-02

[MCQ]

1. Consider the following program:

```
#include<stdio.h>

#include<string.h>
int main()
{
char a[]="GATE_Wallah";
printf("%s\t", &5[a]);
printf("%s", (a+5));
return 0;
}
```

The output is-

- (a) Runtime Error
- (b) Wallah Wallah
- (c) _Wallah _Wallah
- (d) Compilation Error

[MCQ]

2. Consider the following program:

```
#include<stdio.h>
#include<string.h>
int main()
{
char s[5];
s="GATE";
printf("%s",s);
return 0;
}
```

The output is-

- (a) GATE
- (b) G
- (c) NULL
- (d) Compiler Error

[NAT]

3. Consider the following program:

```
#include<stdio.h>
#include<string.h>
```

```
int main()
{
char *p="abcd";
char *q="acd";
int a;
a=strcmp(p,q)?strlen(p):strlen(q);
printf("%d", a);
return 0;
}
```

The output is _____.

[MCQ]

4. Consider the following program:

```
#include<stdio.h>
#include<string.h>
int main()
{
char * s[5]={"CS", "MECH", "ECE",
"ELECTRICAL", "CIVIL"};
char ** p[5]= {s+2, s+4, s+1, s+3, s};
char ***q=p;
q=q+3;
printf("%s",q[-2][-1]);
q=q-2;
printf("%c",***q+++1);
return 0;
}
```

The output printed is-

- (a) CSF
- (b) ELECTRICALD
- (c) CSD
- (d) CIVILF

[MCQ]

5. Consider the following program:

```
#include<stdio.h>
#include<string.h>
int main()
{
char a[]="GATE2024";
```

```

char b[9];
strcpy(b, a);
printf("%s%d%d", b, strlen(b), sizeof(b));
return 0;
}

```

The output is –

- (a) Compilation Error
- (b) GATE202488
- (c) GATE202489
- (d) GATE202499

[MCQ]

6. Consider the following program:

```

#include<stdio.h>
#include<string.h>
void func(char *ptr)
{
    if((*ptr)!='\0'){
        printf("%c", *ptr);
        func(ptr+2);
    }
}
int main()
{
    func("GATEWallah");
    return 0;
}

```

The output is _____

- (a) GTWla
- (b) GTWlh
- (c) GATEWallah
- (d) None of the above

[MCQ]

7. Consider the following program:

```

#include<stdio.h>
#include<string.h>
int main()
{
    char a[]="GATEWallah";
    char *p;
    p=a;
    p+=4;
    *p='\0';
    printf("%s",p);
    return 0;
}

```

```

}

```

The output is-

- (a) No output
- (b) Wallah
- (c) \0allah
- (d) GATE\0allah

[NAT]

8. Consider the following program:

```

#include<stdio.h>
#include<string.h>
int main()
{
    char *a[]={"GATE", "Parakram", "Wallah",
    "Shreshth"};
    char **b[]={a+2, a+3, a+1, a};
    char ***c=b;
    printf("%s", *--*++c+3);
    printf("%s", **++c);
    return 0;
}

```

The length of the output string is _____.

Answer Key

1. (b)
2. (d)
3. (4)
4. (b)

5. (c)
6. (a)
7. (a)
8. (11)



Hints and solutions

1. (b)

%s takes the start address and prints until it reaches NULL character.

```
char a[]="GATE_Wallah";
printf("%s\t", &a[5]); Address passed is (a+5).
"Wallah" is printed.
printf("%s", (a+5)); "Wallah" is printed.
```

Output: Wallah Wallah

2. (d)

s means the base address of the string.
s="GATE"; // A string "GATE" is being assigned to an address. Incompatible assignment.

Output: Compilation Error

3. (4)

strcmp(p,q) compares the two strings by taking the base addresses as parameters. It returns 0 only if they are equal.

The strings "abcd" and "acd" are not equal. So strcmp(*p,*q) returns a non-zero value.

Since, the condition is TRUE, strlen(p) is assigned to a.
a=4.

4. (b)

100	400
C S	E L E C T R I C A L

200	300	500
M E C H	E C E	C I V I L

S	100	200	300	400	500
	600	604	608	612	616

p	608	616	604	612	600
	700	704	708	712	716

q	700	712	704
---	----------------	----------------	-----

q = q + 3

```
printf("%s", q[-2] [-1]);
```

⇒ $*(q - 2) - 1$

⇒ $*(704 - 1)$

⇒ $*(616 - 1)$

⇒ $*(612)$

⇒ 400 ⇒ ELECTRICAL

q = q - 2; // q = 712 - 2 * 4 = 704

```
printf("%c", ***q++ + 1);
```

⇒ $***704 + 1$

⇒ $**616 + 1$

⇒ $*500 + 1$

⇒ C + 1

⇒ D

∴ Output is: ELECTRICALD

5. (c)

```
char a[]="GATE2024";
char b[9];
strcpy(b, a); // The string a is copied to string b.
printf("%s%d%d", b, strlen(b), sizeof(b));
//strlen(b)=8 and sizeof(b)=9
return 0;
```

Output: GATE202489

6. (a)

10	10	10	10	10	10	10	10	10	10	11
0	1	2	3	4	5	6	7	8	9	0
G	A	T	E	W	a	l	l	a	h	\0

func("GATEWallah"); //Address of "GATEWallah" i.e 100 is passed.

ptr: 100

*ptr or *100==G!=\0

printf("%c", *ptr); //G is printed

func(102) is called. It prints *102 i.e T.

So, similarly, func(104), func(106), func(108), func(110) will be called.

Output: GTWla

7. (a)

10	10	10	10	10	10	10	10	10	10	11
0	1	2	3	4	5	6	7	8	9	0
G	A	T	E	W	a	l	l	a	h	\0

p=100;

p+=4;//p=104

*104='\0'

printf("%s",p);//It will print from 104. 104 contains NULL.

Hence, no output.

8. (11)

10	10	10	10						
0	1	2	3						
G	A	T	E						
20	20	20	20	20	20	20	20	20	
0	1	2	3	4	5	6	7	8	
P	a	r	a	k	r	a	m	\0	
30	30	30	30	30	30	30			
0	1	2	3	4	5	6			
W	a	l	l	a	h	\0			

40	40	40	40	40	40	40	40	40
0	1	2	3	4	5	6	7	8
S	h	r	e	s	h	t	h	\0

a:

50	50	50	51
0	4	8	2

10	20	30	40
0	0	0	0

b:

600	604	608	612
508	512	504	500
c	600 604 608		

printf("%s", *--*++c+3);

//*--*++c+3 = *--*604+3 = *--512+3 = *508+3=300+3=303

// 'lah' is printed.

printf("%s", **++c);/**608=*504=200

// 'Parakram' is printed.

Output: lahParakram

Size of the output string: 11



Any issue with DPP, please report by clicking here:- <https://forms.gle/t2SzQVvQcs638c4r5>

For more questions, kindly visit the library section: Link for web: <https://smart.link/sdfez8ejd80if>



PW Mobile APP: <https://smart.link/7wwosivoicgd4>