# CHAPTER 10
## STRINGS

1. O/p?

```
main()
{
  printf(5+ “Fascimile”);
}
```

a. Error
b. Fascimile
c. mile
d. None of above

**Ans: C**

2. O/p?

```
main()
{
 char str1[]= “Hello”;
 char str2[]= “Hello”;
   if(str1==str2)
       printf(“\n Equal”);
   else
       printf(“\nUnequal”);
}
```

a.Equal
b.Unequal
c. Error
e. None of above

**Ans:  B**

3. O/p?

```
main()
{
    printf(“%c”,”abcdefgh”[4]);
}
```

a. Error
b. d

c. e
d. abcdefgh

**Ans: C**

4. O/p?

```
main()
{
   char str[7]= “Strings”;
    printf(“%s”,str);
}
```

a. Error
b. Strings
c. Can not predict
d. None of above

Ans: C
Here str[] has been declared as a 7 character array into it a 8 character string has been stored. This would result into overwriting of the byte beyond the seventh byte reserved for the array with a ‘\0’. There is always a possibility that something important gets overwritten which would be unsafe.

5. How would you output \n on screen?

Ans: printf(\\n);

6. O/p?

```
main()
{
  char ch=’A’;
  printef(“%d  %d”, sizeof(ch), sizeof(‘A’));
}
```

a. 1 1
b. 1 2
c. 2 2
d. 2 1

Ans: B.

7. O/p?

```
main()
{
   printf(“\n %d  %d  %d”, sizeof(‘3’), sizeof(“3”), sizeof(3));
}
```

a. 1 1 1
b. 2 2 2
c. 1 2 2
d. 1 1 1

Ans: B.

8. Is the following program correct?

```
main()
{
  char *str1= “United”;
  char *str2= “Front”;
  char *str3;
  str3=strcat(str2, str2);
  printf(“\n %s”,str3);
}
```

Ans:NO.
   Since what is present in the memory bruond ‘United’ is not known and we are attaching ‘Front at the end of it, thereby overwriting something.

9. How would you improve the code in Q8 above?

Ans:

```
main()
{
  char str1[15]= “United”;
  char *str2= “Front”;
  char *str3;
  str3= strcat(str1, str2);
  printf(“\n %s”,str3);
}
```

10. In the following code which function would get called, the user defined strcpy () or the one in the standard library?

```
main()
{
  char str1[]= “Keep India Beautiful… emigrate!”;
  char str2[40];
  strcpy(str2, str1);
  printf(“\n %s”, str2);
}
strcpy(char *t, char *s)
{
  while(*s)
   {
       *t = *s;
        t++;
        s++;
   }
  *t= ‘\0’;
}
```

Ans: User defined strcpy().

11. Can you compact the code in strcpy() into one line?

Ans:

```
 strcpy( char *t, char *s)
 {
   while (*t++ = *s++);
 }
```

12. O/p?

```
main()
{
   char *str[] = { “Frogs”, “Do”, “Not”, “They”, “Croak!”};
   printf(“%d  %d”, sizeof(str), sizeof(str[0]));
}
```

**Ans: 12  2**

13. How would you find the length of each string in the program above?

Ans:

```
main()
{
  char *str[] = {“Frogs”, “Do”, “Not”, “They”, “Croak!”};
  int i;
       for(i=0; i<=4; i++)
          printf(“\n %s  %d”, str[i], strlen(str[i]));
}
```

14. What is the difference in the following declarations

```
char *p = “Samuel”;
char a[] = “Samuel”;
```

Ans: Here a is an array big enough to hold the message and the ‘\0’ following the message. Individual characters within the array can be changed but the address of the array would remain same.

On the other hand p is a pointer, initialized to point to a string constant. The pointer p may be modified to point to another string, but if you attempt to modify the string at which p is pointing the result is undefined.

15. While handling the string do we always have to process it character by character or there exists a method to process the entire string as one unit.

Ans: A string can be processed only on a character by character basis.