STRINGS –PREDICT THE OUTPUT

Strings and Pointers

PREDICT THE OUTPUT

str[7] is same as *(str+7) .
7[str] how the compiler parse it is again as *(7+str) which is same as str[7]

```
#include<stdio.h>
int main()
{
    char str[] = "Amrita University";
    printf("%s %s %s\n", &str[7], &7[str], str+7);
    printf("%c %c %c\n", *(str+7), str[7], 7[str]);
    return 0;
}
```

Second Parameter of printf("%s",str) is str which is starting address of string. So &str[7] and &7[str] and str+7 will all give address of U. And anyways '\0' is there at the end of university. So it completes a string

Output :

```
University University University
U  U  U
```

PREDICT THE OUTPUT

```
#include "stdio.h"
int main()
{
  char str[] = { 'A', 'B', 'C', 'D','\0' };
   printf("%s \n",str);
   char *p = &str[1];
   p++;
   printf("%c ", *p);
   p++;
   printf("%c ", *p);
}
```

Though initialised, str is a complete string since it has ‘\0’ at end .hence can be printed using %s

P points to B. Then incremented once and hence points to C .*p is C. Once again incremented.Points to D.*p is D

Output

```
ABCD
C   D
```

PREDICT THE OUTPUT

```
#include <stdio.h>
int fun(char *str1)
{ printf("%c ",*str1);
  char *str2 = str1;
  while(*++str1);
  return (str1-str2);
}

int main()
{
  char *str = "Amrita";
  printf("%d", fun(str));
  return 0;
}
```

//Display A

//Str2 point to Amrita

//The statement while(*++str1); increments str1 till ‘\0’ is reached.
//str1 is incremented by 6. pls note no stmts inside while
//difference between str2 and str1 is returned which is 6

OUTPUT:A 6

Predict the output

```c
#include <stdio.h>
int fun(char *p)
{
    if (p == NULL || *p == '\0') return 0;
    int current = 1, i = 1;
    while (*(p+current))
    {
        if (p[current] != p[current-1])
        {
            p[i] = p[current];
            i++;
        }
        current++;
    }
    *(p+i)='\0';
    return i;
}
```

```c
int main()
{
    char str[] = "RRREEERPPPXXXP";
    fun(str);
    puts(str);
    return 0;
}
```

OUTPUT: RERPXP

P is not null, it has start address of string. Current start with 1. everytime check happens whether p[current] and p[current-1] are same. ie whether consecutives are same .If not same then only p gets copied with current character. A new index i starting with 1 act as index for doing the new updation in p. and at the end p[i] is added with '\0'.There ends the string.

Namah shivaya