

#### Lab 4: Assignment

1) What will be the output of the following program? Note down your understanding of every program, in few sentences.

a)

Answer:

The output of above source cod will be 30 because the value of I is 3 and value of \*j is also 3 since the j is the address of I.

i=3

j=address of i

\*j=value of j which is 3

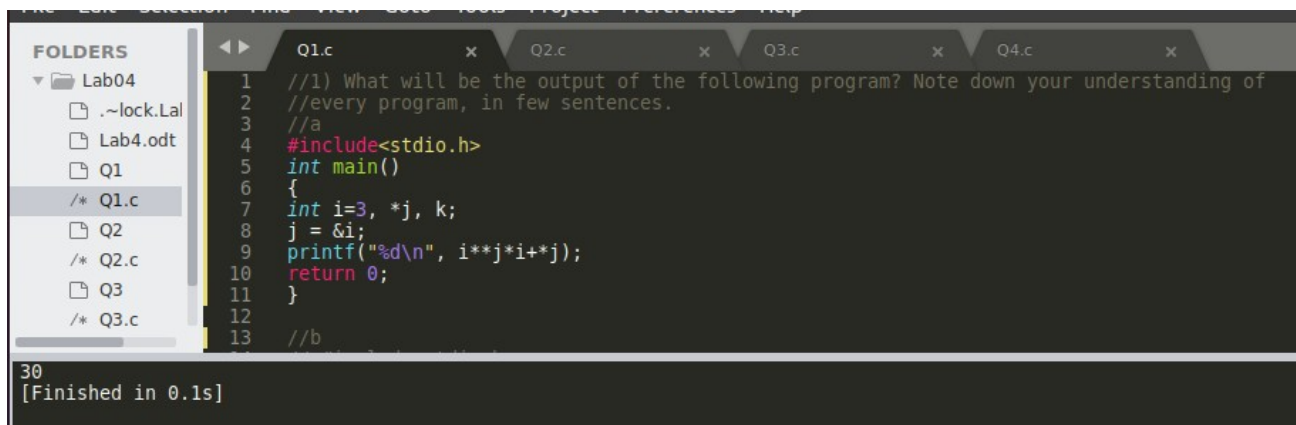
now,

=> i\*\*j\*i+\*j

=>3\*3\*3+3

=>30

Output:



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a folder named 'Lab04' containing files like 'Q1.c', 'Q2.c', 'Q3.c', and 'Q4.c'. The code editor shows the content of 'Q1.c', which is a C program. The program starts with a comment asking for the output of a program. It includes `<stdio.h>` and defines a `main` function. Inside `main`, it declares `int i=3, *j, k;`, assigns `j = &i;`, and prints `i**j*i+*j` using `printf`. The output of the program is shown in a terminal window at the bottom, which displays '30' and '[Finished in 0.1s]'.

```
1 //1) What will be the output of the following program? Note down your understanding of
2 //every program, in few sentences.
3 //a
4 #include<stdio.h>
5 int main()
6 {
7     int i=3, *j, k;
8     j = &i;
9     printf("%d\n", i**j*i+*j);
10    return 0;
11 }
12
13 //b
```

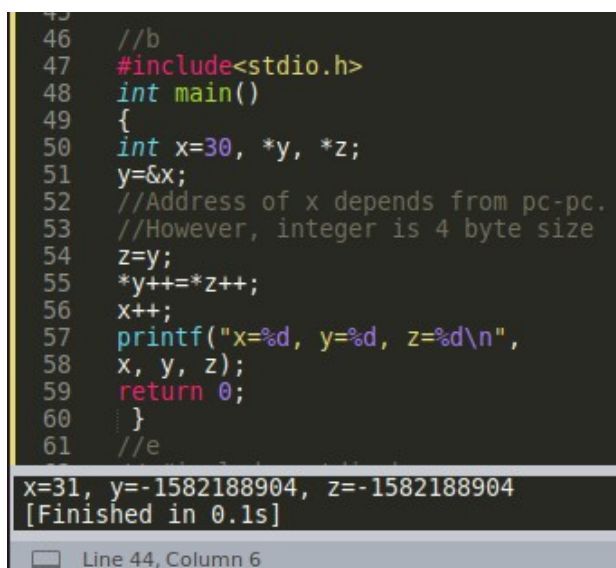
30  
[Finished in 0.1s]

b)

Answer:

In this question x=31 because the x is incremented, y is the address of x that is also incremented by 1 and output of z= to the output of y since z=y and it will be incremented by 1. Hence the value of y and z be equal.

Output:



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a folder named 'Lab04' containing files like 'Q1.c', 'Q2.c', 'Q3.c', and 'Q4.c'. The code editor shows the content of 'Q1.c', which is a C program. The program starts with a comment asking for the output of a program. It includes `<stdio.h>` and defines a `main` function. Inside `main`, it declares `int x=30, *y, *z;`, assigns `y = &x;`, and increments `x` by 1. It then assigns `z = y;` and increments `*y` by `*z++`. Finally, it prints `x, y, z` using `printf`. The output of the program is shown in a terminal window at the bottom, which displays 'x=31, y=-1582188904, z=-1582188904' and '[Finished in 0.1s]'.

```
46 //b
47 #include<stdio.h>
48 int main()
49 {
50     int x=30, *y, *z;
51     y=&x;
52     //Address of x depends from pc-pc.
53     //However, integer is 4 byte size
54     z=y;
55     *y++=*z++;
56     x++;
57     printf("x=%d, y=%d, z=%d\n",
58           x, y, z);
59     return 0;
60 }
61 //e
```

x=31, y=-1582188904, z=-1582188904  
[Finished in 0.1s]

Line 44, Column 6

c)

Answer:

Here  $i=8$  and  $p$  is the address of  $i$  and output for  $p^*$  will be 8 since the  $p$  is pointed towards  $i$ .  $q$  is the address of  $p$  and  $p$  points towards  $i$  so the outcome of  $**p$  will be 8 and  $r$  is the address of  $q$  where  $q$  is pointed towards  $p$  and  $p$  is  $i$  (pointed towards  $i$ ) the outcome  $***r$  will be 8.

Outcome:

```
29 //c
30 #include<stdio.h>
31 int main()
32 {
33     int ***r, **q, *p, i=8;
34     p = &i;
35     q = &p;
36     r = &q;
37     printf("%d, %d, %d\n", *p,
38           **q, ***r);
39     return 0;
40 }
41
```

8, 8, 8  
[Finished in 0.1s]

d)

Output:

```
43 #include<stdio.h>
44 void fun(void *p);
45 int i;
46 int main()
47 {
48     void *vptr;
49     vptr = &i;
50     fun(vptr);
51     return 0;
52 }
53 void fun(void *p)
54 {
55     int **q;
56     q = (int**) &p;
57     printf("%d\n", **q);
58 }
59
```

0  
[Finished in 0.1s]

e)

Output:

```
62 //
63 #include <stdio.h>
64 int main()
65 {
66     int *ptr;
67     int x;
68     ptr = &x;
69     *ptr = 0;
70     printf(" x = %d\n", x);
71     printf(" *ptr = %d\n", *ptr);
72     *ptr += 5;
73     printf(" x = %d\n", x);
74     printf(" *ptr = %d\n", *ptr);
75     (*ptr)++;
76     printf(" x = %d\n", x);
77     printf(" *ptr = %d\n", *ptr);
78     return 0;
79 }
```

x = 0  
\*ptr = 0  
x = 5  
\*ptr = 5  
x = 6  
\*ptr = 6  
[Finished in 0.2s]

f)

Output:

```
78 // }
79
80 //f
81 #include<stdio.h>
82 int main()
83 {
84     char *str;
85     str = "%s";
86     printf(str, "K\n");
87     return 0;
88 }
```

K  
[Finished in 0.1s]

2) Write a C Program to swap 4-different elements using Call by Reference.

Answer:

Source code:

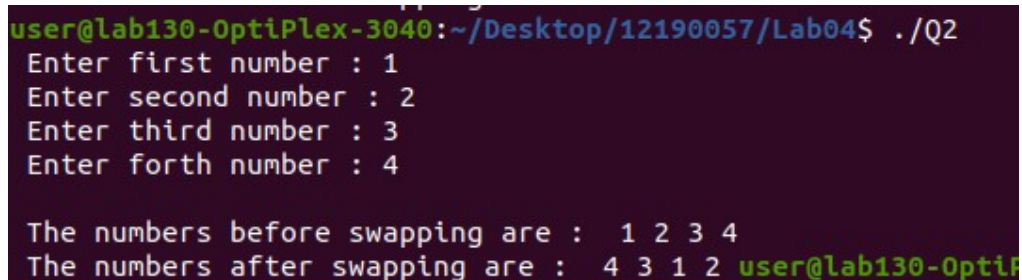
```
#include <stdio.h>

int swap(int *a,int *b,int *c,int *d)
{
    int num;
    num=*a;
    *a=*d;
    *d=*b;
    *b=*c;
    *c=num;
}

int main()
{
    int num1,num2,num3,num4;
    printf(" Enter first number : ");
    scanf("%d",&num1);
    printf(" Enter second number : ");
    scanf("%d",&num2);
    printf(" Enter third number : ");
    scanf("%d",&num3);
    printf(" Enter forth number : ");
    scanf("%d",&num4);

    printf("\n The numbers before swapping are : ");
    printf(" %d %d %d %d",num1,num2,num3,num4);
    swap(&num1,&num2,&num3,&num4);
    printf("\n The numbers after swapping are : ");
    printf(" %d %d %d %d ",num1,num2,num3,num4);
    return 0;
}
```

Output:



```
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$ ./Q2
Enter first number : 1
Enter second number : 2
Enter third number : 3
Enter forth number : 4

The numbers before swapping are : 1 2 3 4
The numbers after swapping are : 4 3 1 2 user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$
```

3) WAP a program to find if the Year entered by the user through keyboard is a leap year or not.  
Apply Call by Reference concept.

Answer:

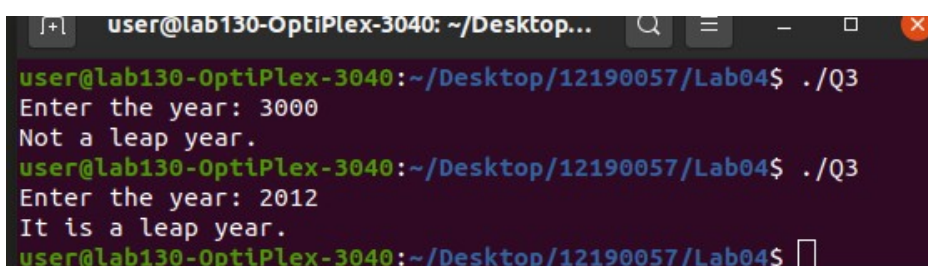
Source code

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

```
int leapYear(int *check, int *x, int *y, int *z){  
    if (*check % *x == 0){  
        if (*check % *y == 0){  
            if (*check % *z == 0){  
                printf("It is a leap year.\n");  
            }  
            else{  
                printf("Not a leap year.\n");  
            }  
        }  
        else if (*check % *y != 0){  
            printf("It is a leap year.\n");  
        }  
        else{  
            printf("It is not a leap year.\n");  
        }  
    }  
    else if(*check % *y == 0){  
        if (*check % *z == 0){  
            printf("It is a leap year.\n");  
        }  
        else{  
            printf("It is not a leap year.\n");  
        }  
    }  
    else{  
        printf("It is not a leap year.\n");  
    }  
}
```

```
int main(){  
    int year;  
    int a = 4;  
    int b = 100;  
    int c = 400;  
    printf("Enter the year: ");
```

Output:



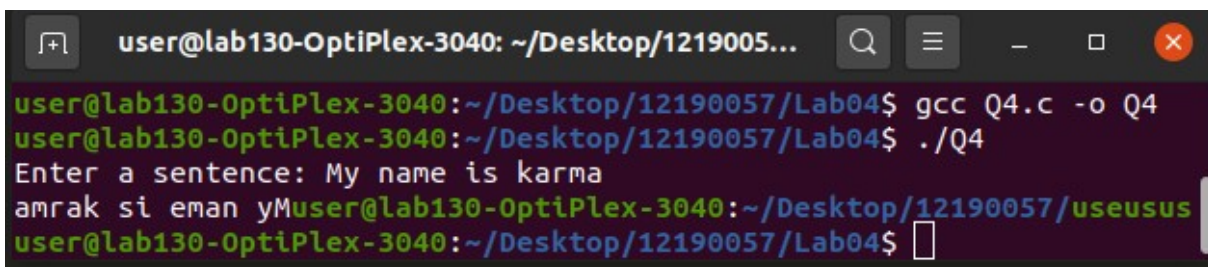
```
user@lab130-OptiPlex-3040: ~/Desktop...  
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$ ./Q3  
Enter the year: 3000  
Not a leap year.  
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$ ./Q3  
Enter the year: 2012  
It is a leap year.  
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$
```

Question 4:

```
#include <stdio.h>
void reverseSentence();
int main()
{
    printf("Enter a sentence: ");
    reverseSentence();
    return 0;
}
void reverseSentence(){
    char c;

    scanf("%c", &c);
    if(c != '\n'){
        reverseSentence();
        printf("%c", c);
    }
}
```

Output:

A terminal window with a dark background and light-colored text. The window title is "user@lab130-OptiPlex-3040: ~/Desktop/1219005...". The prompt is "user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04\$". The first command is "gcc Q4.c -o Q4". The second command is "./Q4". The output is "Enter a sentence: My name is karma". The next line shows the reversed string "amrak si eman yM". The prompt is "user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04\$".

```
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$ gcc Q4.c -o Q4
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$ ./Q4
Enter a sentence: My name is karma
amrak si eman yMuser@lab130-OptiPlex-3040:~/Desktop/12190057/useusus
user@lab130-OptiPlex-3040:~/Desktop/12190057/Lab04$
```