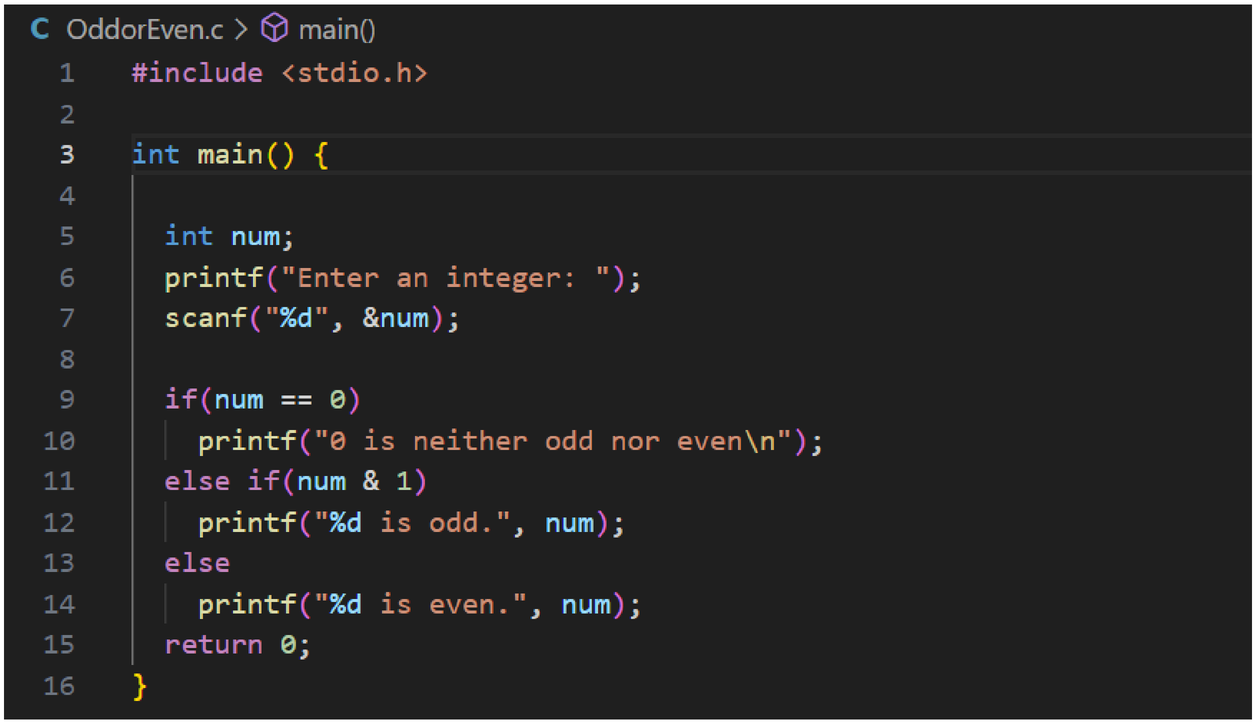
ADVANCED C PROGRAMMING

Module – 1 Assessment

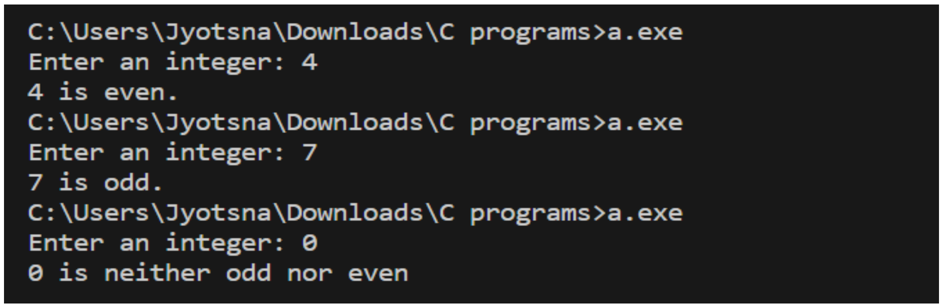
1. **Write a C program to determine whether the given number is odd or even using Bitwise operators.**



OddorEven.c > main()

```c
#include <stdio.h>

int main() {

    int num;
    printf("Enter an integer: ");
    scanf("%d", &num);

    if(num == 0)
        printf("0 is neither odd nor even\n");
    else if(num & 1)
        printf("%d is odd.", num);
    else
        printf("%d is even.", num);
    return 0;
}
```

OUTPUT:



```
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter an integer: 4
4 is even.
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter an integer: 7
7 is odd.
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter an integer: 0
0 is neither odd nor even
```

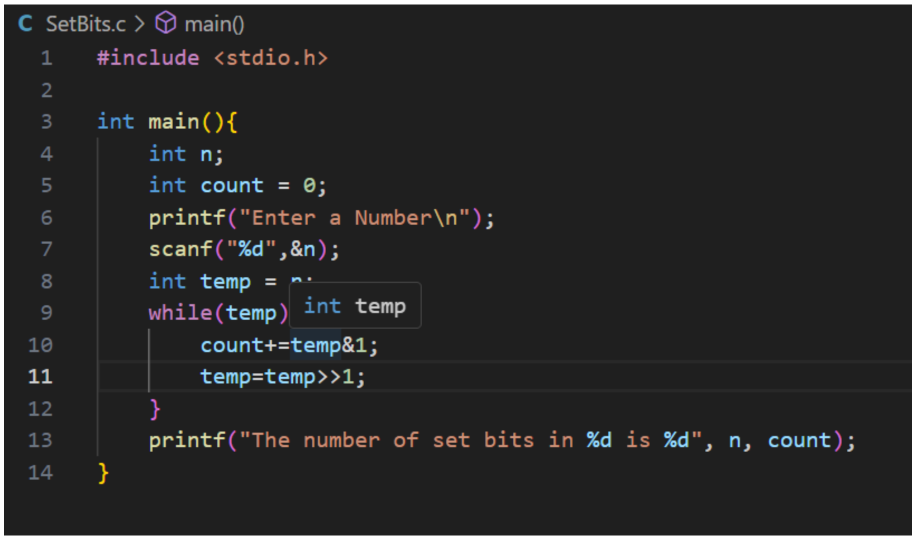
2. Write a C program to count the number of bits set in a number

Input:

144

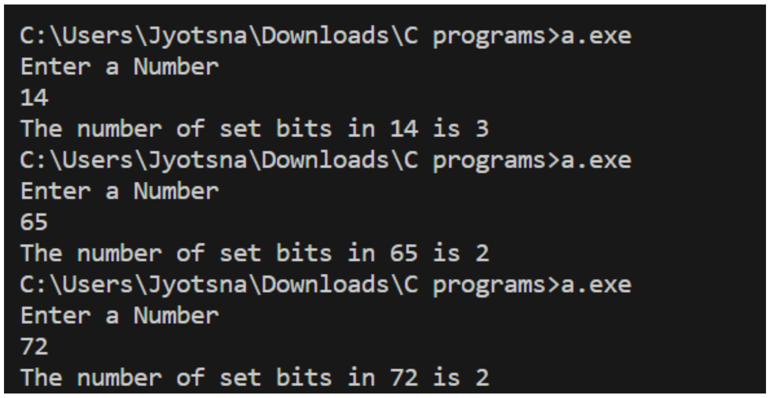
Output:

Count of Set bits: 2



```c
#include <stdio.h>

int main(){
    int n;
    int count = 0;
    printf("Enter a Number\n");
    scanf("%d",&n);
    int temp = n;
    while(temp){
        count+=temp&1;
        temp=temp>>1;
    }
    printf("The number of set bits in %d is %d", n, count);
}
```

OUTPUT:



```
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter a Number
14
The number of set bits in 14 is 3
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter a Number
65
The number of set bits in 65 is 2
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter a Number
72
The number of set bits in 72 is 2
```

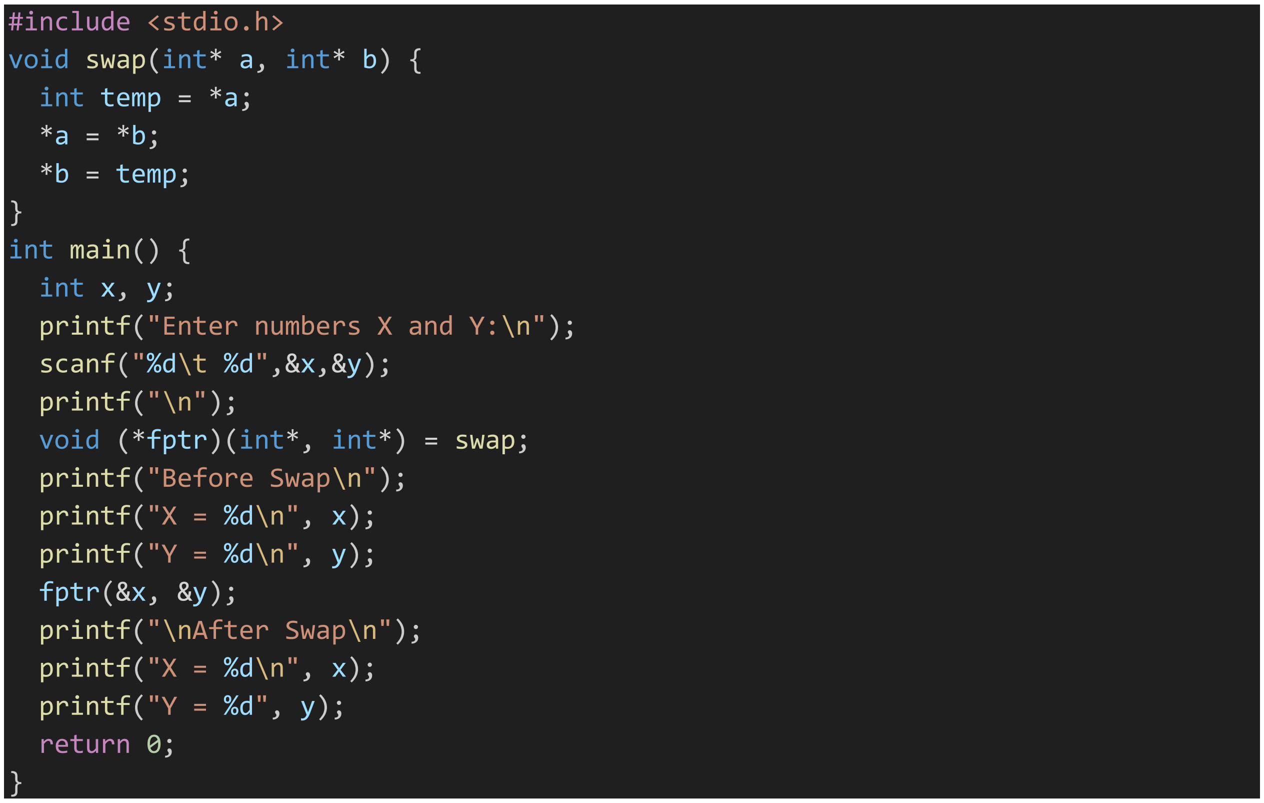
3. Write a C program to swap two numbers. Use a function pointer to do this operation.

Input:

84 25

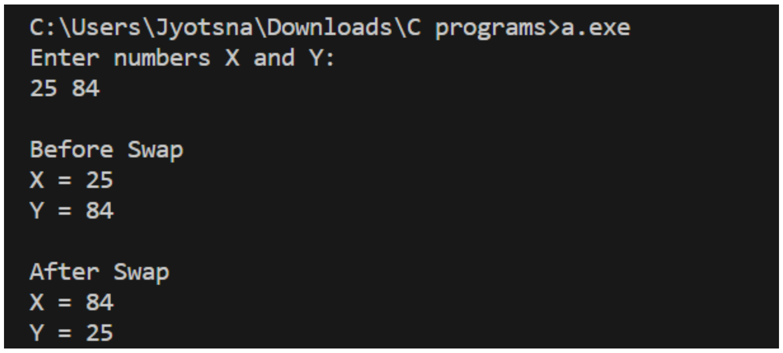
Output:

25 84



```c
#include <stdio.h>
void swap(int* a, int* b) {
  int temp = *a;
  *a = *b;
  *b = temp;
}
int main() {
  int x, y;
  printf("Enter numbers X and Y:\n");
  scanf("%d\t %d",&x,&y);
  printf("\n");
  void (*fptr)(int*, int*) = swap;
  printf("Before Swap\n");
  printf("X = %d\n", x);
  printf("Y = %d\n", y);
  fptr(&x, &y);
  printf("\nAfter Swap\n");
  printf("X = %d\n", x);
  printf("Y = %d", y);
  return 0;
}
```

OUTPUT:



```
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter numbers X and Y:
25 84

Before Swap
X = 25
Y = 84

After Swap
X = 84
Y = 25
```

4. Write an equivalent pointer expression for fetching the value of array element a[i][j][k][2]

Solution: *(*(*(*(a + i) + j) + k) + 2)

5. Write a C program to Multiply two matrix (n*n) using pointers.

Input:

Size of Row: 3

Size of Column: 3

Matrix 1:

2 3 4

5 6 7

8 9 1

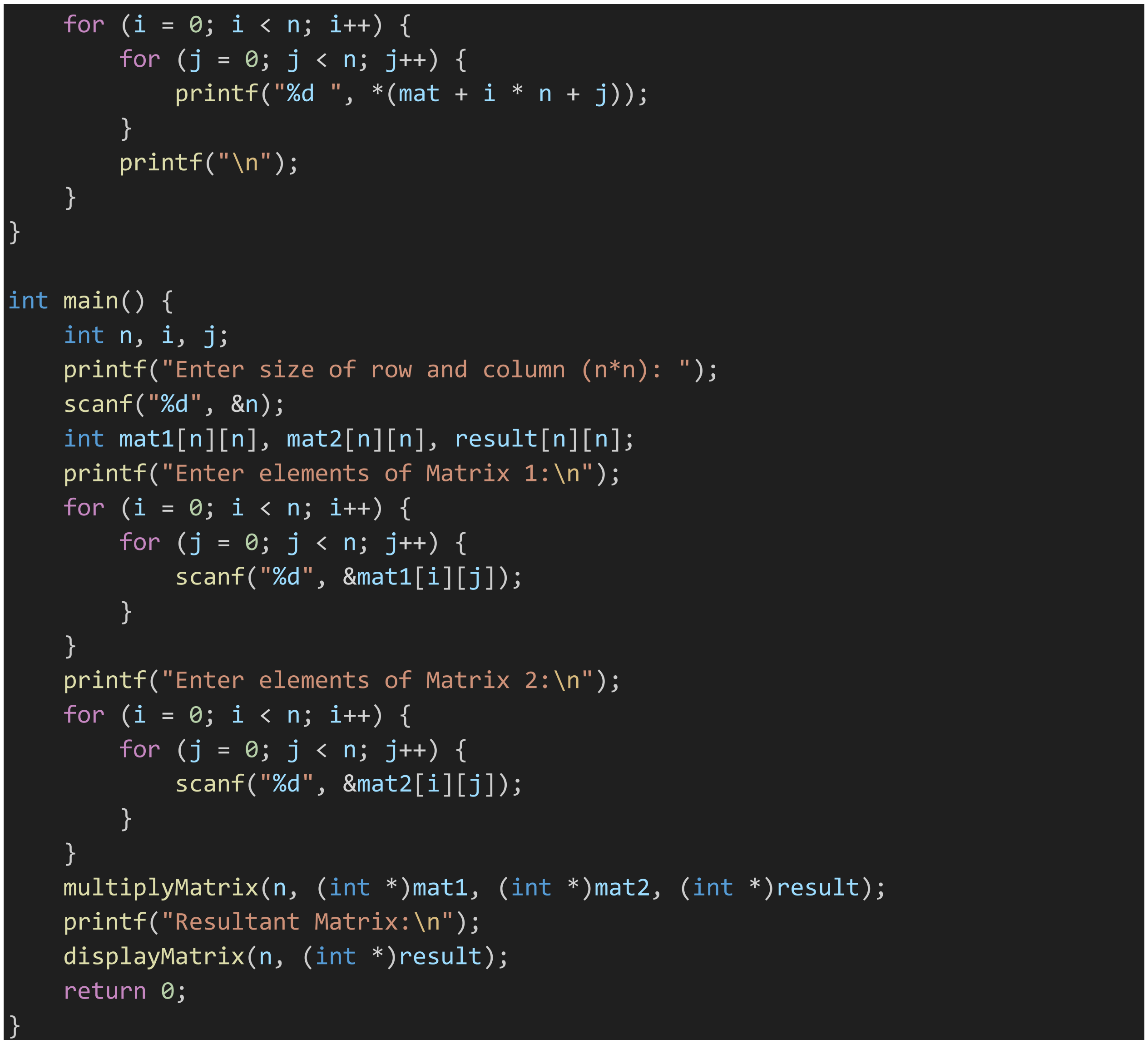
Matrix 2:

9 8 7

6 5 4

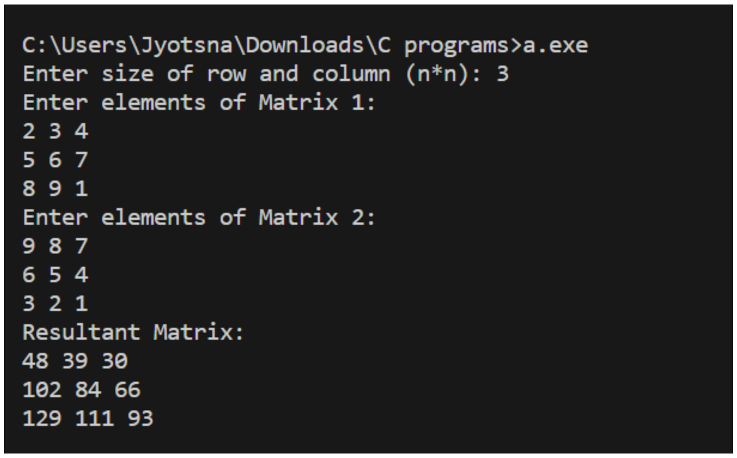
3 2 1

```c
#include <stdio.h>

void multiplyMatrix(int n, int *mat1, int *mat2, int *result) {
    int i, j, k;

    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            *(result + i * n + j) = 0;

            for (k = 0; k < n; k++) {
                *(result + i * n + j) += *(mat1 + i * n + k) * *(mat2 + k * n +
j);
            }
        }
    }
}

void displayMatrix(int n, int *mat) {
    int i, j;
```



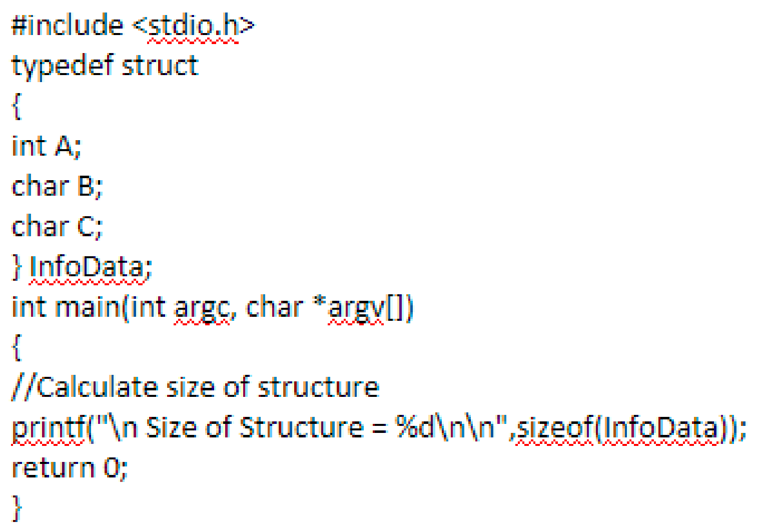
```c
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            printf("%d ", *(mat + i * n + j));
        }
        printf("\n");
    }
}

int main() {
    int n, i, j;
    printf("Enter size of row and column (n*n): ");
    scanf("%d", &n);
    int mat1[n][n], mat2[n][n], result[n][n];
    printf("Enter elements of Matrix 1:\n");
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            scanf("%d", &mat1[i][j]);
        }
    }
    printf("Enter elements of Matrix 2:\n");
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            scanf("%d", &mat2[i][j]);
        }
    }
    multiplyMatrix(n, (int *)mat1, (int *)mat2, (int *)result);
    printf("Resultant Matrix:\n");
    displayMatrix(n, (int *)result);
    return 0;
}
```

OUTPUT:



```
C:\Users\Jyotsna\Downloads\C programs>a.exe
Enter size of row and column (n*n): 3
Enter elements of Matrix 1:
2 3 4
5 6 7
8 9 1
Enter elements of Matrix 2:
9 8 7
6 5 4
3 2 1
Resultant Matrix:
48 39 30
102 84 66
129 111 93
```

6. Find the output of the following // Consider the compiler is 32-bit machine

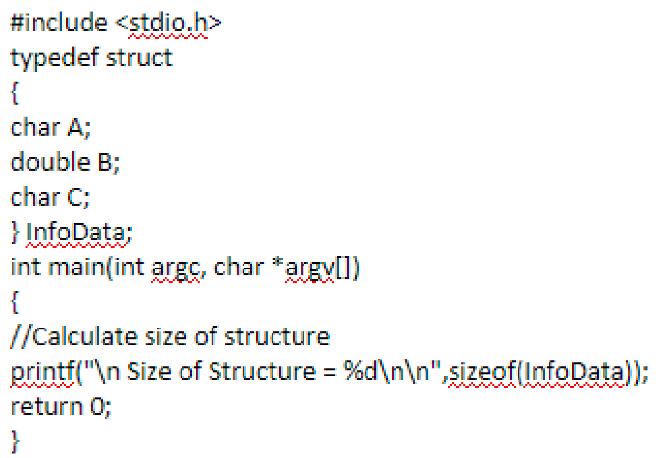


```
#include <stdio.h>
typedef struct
{
int A;
char B;
char C;
} InfoData;
int main(int argc, char *argv[])
{
//Calculate size of structure
printf("\n Size of Structure = %d\n\n",sizeof(InfoData));
return 0;
}
```

OUTPUT:

8 bytes.

7. Find the output of the following // Consider the compiler is 32-bit machine

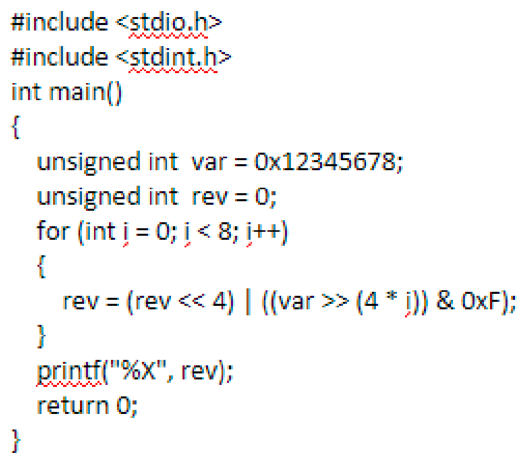


```
#include <stdio.h>
typedef struct
{
char A;
double B;
char C;
} InfoData;
int main(int argc, char *argv[])
{
//Calculate size of structure
printf("\n Size of Structure = %d\n\n",sizeof(InfoData));
return 0;
}
```

OUTPUT:

16 bytes.

8. Find the output of the following // Consider the compiler is 32-bit machine



```
#include <stdio.h>
#include <stdint.h>
int main()
{
   unsigned int  var = 0x12345678;
   unsigned int  rev = 0;
   for (int i = 0; i < 8; i++)
   {
      rev = (rev << 4) | ((var >> (4 * i)) & 0xF);
   }
   printf("%X", rev);
   return 0;
}
```

OUTPUT:

87654321