TAMMALI KARTHIK BU21EECE0100164

HANDS ON ACTIVITY EMBEDDED SYSYTEMS.

1] Write a program to count no. of bits which are set in given binary pattern .

Code::

```
#include <stdio.h>
int countSetBits(unsigned int num) {
int count = 0;
while (num) {
count += num & 1; num >>= 1;
} return count; }
int main() {
unsigned int num = 0b10101010;
printf("Number of set bits: %d\n", countSetBits(num));
return 0; }
```

Output: Number of set bits: 4

2] Write a program to set 5th and 12th bits in a 16-bit unsigned integer

Code:

```
#include <stdio.h>
unsigned int setBits(unsigned int num, int pos1, int pos2) {
unsigned int mask = (1 << pos1) | (1 << pos2);
return num | mask;
}
int main() {
unsigned int num = 0b00000000; num = setBits(num, 5, 12);
printf("Modified number: %d\n", num);
return 0;
}
```

Output: Modified number: 4864

3] Write a program to clear 6th and 19th bits in a 32-bit unsigned integer.

Code:

```
#include <stdio.h>
unsigned int clearBits(unsigned int num, int pos1, int pos2) {
unsigned int mask = ~(1 << pos1) & ~(1 << pos2);
return num & mask;
}
int main() { unsigned int num = 0b11111111111111111;
num = clearBits(num, 6, 19);
printf("Modified number: %u\n", num);
return 0;
}
```

Output: Modified number: 524287

4] Write a program to flip even positioned bits in a 16-bit unsigned integer

Code:

```
#include <stdio.h>
unsigned int flipEvenBits(unsigned int num) {
unsigned int mask = 0xAAAA; // Binary pattern with even bits set
return num ^ mask;
}
int main() {
unsigned int num = 0b1010101010101010; // Example 16-bit unsigned
integer
num = flipEvenBits(num);
printf("Modified number: %d\n", num);
return 0;
```

Output: Modified number: 2730

5] Given an unsigned 32-bit integer holding packed IPv4 address, convert it into "a. b. c. d" format.

Code:

```
#include <stdio.h>
int countSetBits(unsigned int num) {
int count = 0;
while (num) {
count += num & 1;
num >>= 1;
}
return count;
}
int main() {
unsigned int num = 0b10101010; // Example binary pattern
printf("Number of set bits: %d\n", countSetBits(num));
return 0;
}
```

Output: Number of set bits: 4

6] Convert MAC address into 48-bit binary pattern

Code:

```
#include <stdio.h>
void unpackIPAddress(unsigned int ip) {
int a, b, c, d;
a = (ip >> 24) & 255;
b = (ip >> 16) & 255;
c = (ip >> 8) & 255;
d = ip & 255;
printf("Unpacked IP address: %d.%d.%d.%d\n", a, b, c, d);
}
int main() {
unsigned int packedIP = 3232235777; // Example packed IP address
```

```
unpackIPAddress(packedIP);
return 0;
```

Output: Unpacked IP address: 192.168.1.1

7] Convert 48-bit binary pattern as MAC address

Code:

```
#include <stdio.h>
void macToBinaryPattern(char *mac) {
unsigned long long int binary = 0;
sscanf(mac, "%2hhx:%2hhx:%2hhx:%2hhx:%2hhx:%2hhx", (unsigned char
*)&binary,
(unsigned char *)&binary + 1, (unsigned char *)&binary + 2,
(unsigned char *)&binary + 3, (unsigned char *)&binary + 4,
(unsigned char *)&binary + 5);
printf("Binary pattern: %llx\n", binary);
}
int main() {
char mac[] = "12:34:56:78:9a:bc"; // Example MAC address
macToBinaryPattern(mac);
return 0;
}
```

Output: Binary pattern: 123456789abc

8] Convert 48-bit binary pattern to MAC address.

Code:

```
#include <stdio.h>
void binaryPatternToMac(unsigned long long int binary)
{
printf("MAC address: %02llx:%02llx:%02llx:%02llx:%02llx:%02llx\n",
(binary >> 40) & 0xFF, (binary >> 32) & 0xFF, (binary >> 24) & 0xFF,
(binary >> 16) & 0xFF, (binary >> 8) & 0xFF, binary & 0xFF);
}
int main() {
```

```
unsigned long long int binary = 0x123456789abc; // Example binary pattern
binaryPatternToMac(binary);
return 0;
}
```

Output: MAC address: 12:34:56:78:9a:bc