1. A. <https://www.javatpoint.com/kali-linux-installation>

B. <https://www.geeksforgeeks.org/how-to-install-virtual-box-in-kali-linux/>

1. <https://www.geeksforgeeks.org/linux-directory-structure/>
2. A. <https://www.javatpoint.com/linux-commands>

B. <https://www.redhat.com/sysadmin/introduction-vi-editor>

1. Theory related to 10 to 12 Networking devices(8 Network device write in 2.3 lecture in class). (If you know about create topology then 1 topology)[Cisco packet Tracer]
2. **Hello print program**:

#include <stdio.h>

#include <unistd.h>

#define MSGSIZE 16

char\* msg1 = "hello friends";

char\* msg2 = "hello, world #2";

char\* msg3 = "hello, world #3";

int main()

{

char inbuf[MSGSIZE];

int p[2], i;

if (pipe(p) < 0)

exit(1);

/\* continued \*/

/\* write pipe \*/

write(p[1], msg1, MSGSIZE);

write(p[1], msg2, MSGSIZE);

write(p[1], msg3, MSGSIZE);

for (i = 0; i < 3; i++) {

/\* read pipe \*/

read(p[0], inbuf, MSGSIZE);

printf("% s\n", inbuf);

}

return 0;

}

1. **Character count program in string:**

|  |  |
| --- | --- |
|  | #include <stdio.h>  #include <string.h>    int main()  {      char s[1000],c;      int i,count=0;        printf("Enter  the string : ");      gets(s);      printf("Enter character to be searched: ");      c=getchar();        for(i=0;s[i];i++)      {       if(s[i]==c)       {            count++;  }  }    printf("character '%c' occurs %d times \n ",c,count);            return 0;  } |

1. **Bits count program stuff :**

#include <stdio.h>

int countSetBits(int n) {

int count = 0;

while (n) {

count += n & 1;

n >>= 1;

}

return count;

}

int main() {

int num;

printf("Enter an integer: ");

scanf("%d", &num);

int result = countSetBits(num);

printf("Number of set bits in %d: %d\n", num, result);

return 0;

}

1. **Perform a GNU C program to generate frames from sender’s message by splitting message by given frame-length.**

#include <stdio.h>

#include <string.h>

#define MAX\_MESSAGE\_LENGTH 1000

void generateFrames(char \*message, int frameLength) {

int messageLength = strlen(message);

int numFrames = (messageLength + frameLength - 1) / frameLength; // Calculate the number of frames needed

int i, j;

printf("Frames:\n");

for (i = 0; i < numFrames; i++) {

printf("Frame %d: ", i + 1);

for (j = 0; j < frameLength && (i \* frameLength + j) < messageLength; j++) {

printf("%c", message[i \* frameLength + j]);

}

printf("\n");

}

}

int main() {

char message[MAX\_MESSAGE\_LENGTH];

int frameLength;

printf("Enter the message: ");

fgets(message, sizeof(message), stdin);

message[strcspn(message, "\n")] = '\0'; // Remove trailing newline

printf("Enter the frame length: ");

scanf("%d", &frameLength);

generateFrames(message, frameLength);

return 0;

}

1. **II) Character Stuffing Program :**

#include <stdio.h>

#include <string.h>

#define MAX\_FRAME\_SIZE 100

void characterStuffing(char\* input, char\* stuffed, char delimiter) {

int i, j = 0;

stuffed[j++] = delimiter; // Start and end delimiter

for (i = 0; i < strlen(input); i++) {

if (input[i] == delimiter) {

stuffed[j++] = delimiter; // Escape the delimiter

stuffed[j++] = delimiter; // Duplicate the delimiter

} else {

stuffed[j++] = input[i];

}

}

stuffed[j++] = delimiter; // End delimiter

stuffed[j] = '\0'; // Null terminator

}

int main() {

char input[MAX\_FRAME\_SIZE];

char stuffed[MAX\_FRAME\_SIZE \* 2]; // Maximum possible stuffed frame size

char delimiter;

printf("Enter the frame: ");

fgets(input, sizeof(input), stdin);

input[strcspn(input, "\n")] = 0; // Remove newline character

printf("Enter the delimiter character: ");

delimiter = getchar();

getchar(); // Consume newline character

characterStuffing(input, stuffed, delimiter);

printf("Stuffed frame: %s\n", stuffed);

return 0;

}

1. **Byte Stuffing :**

#include <stdio.h>

#include <string.h>

int main() {

char frame[50][50], str[50][50];

char flag[10];

strcpy(flag, "flag");

char esc[10];

strcpy(esc, "esc");

int i, k = 0, n;

strcpy(frame[k++], flag);

printf("Enter length of String : \n");

scanf("%d", &n);

printf("Enter the String: ");

getchar(); // to clear the buffer

for (i = 0; i < n; i++) {

fgets(str[i], sizeof(str[i]), stdin);

str[i][strcspn(str[i], "\n")] = '\0'; // remove newline character

}

printf("\nYou entered :\n");

for (i = 0; i < n; i++) {

puts(str[i]);

}

printf("\n");

for (i = 0; i < n; i++) {

if (strcmp(str[i], flag) != 0 && strcmp(str[i], esc) != 0) {

strcpy(frame[k++], str[i]);

} else {

strcpy(frame[k++], esc);

strcpy(frame[k++], str[i]);

}

}

strcpy(frame[k++], flag);

printf("------------------------------\n\n");

printf("Byte stuffing at sender side:\n\n");

printf("------------------------------\n\n");

for (i = 0; i < k; i++) {

printf("%s\t", frame[i]);

}

return 0;

}

1. **Bit Stuffing Program:**

#include <stdio.h>

#include <string.h>

int main() {

char data[100], stuffedData[200];

int i, count = 0, j = 0;

printf("Enter the data: ");

scanf("%s", data);

for(i = 0; i < strlen(data); i++) {

if(data[i] == '1') {

count++;

stuffedData[j++] = data[i];

} else {

count = 0;

stuffedData[j++] = data[i];

}

if(count == 5) {

count = 0;

stuffedData[j++] = '0';

}

}

stuffedData[j] = '\0';

printf("Data after bit stuffing: %s\n", stuffedData);

return 0;

}