Program 1: Character Stuffing

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

int main() {
    char in[50], out[60] = "$";

    printf("Enter data: ");
    scanf("%s", in);

    strcat(out, in);
    strcat(out, "$");

    printf("Stuffed: %s\n", out);
    return 0;
}
```

Program 2: Bit Stuffing

```
#include <stdio.h>

int main() {
    char in[50];
    int c = 0;

    printf("Enter bits: ");
    scanf("%s", in);

    for (int i = 0; in[i]; i++) {
        putchar(in[i]);
        c = (in[i] == '1') ? c + 1 : 0;

        if (c == 5) {
            putchar('0');
            c = 0;
        }
    }
}
```

Program 3: Checksum

```
#include <stdio.h>

int main() {
    int n, x, sum = 0;

    printf("ENTER SIZE OF THE STRING: ");
    scanf("%d", &n);

    printf("ENTER THE ELEMENTS OF THE ARRAY TO CALCULATE CHECKSUM:\n");
    while (n--) {
        scanf("%d", &x);
        sum += x;
    }

    printf("\n*SENDER SIDE*\nSUM IS: %d\nCHECKSUM IS: %d", sum, -sum);

    printf("\n\n*RECEIVER SIDE*\nSUM IS: %d\nCHECKSUM IS: %d",
           sum, sum + (-sum));

    return 0;
}
```

Program 4: Hamming Code

```
#include <stdio.h>

int main() {
    char s[5];
    int d[4], h[7];

    printf("Enter 4 data bits (0/1): ");
    scanf("%s", s);

    for (int i = 0; i < 4; i++)
        d[i] = s[i] - '0';

    h[2] = d[0];
    h[4] = d[1];
    h[5] = d[2];
    h[6] = d[3];

    h[0] = h[2] ^ h[4] ^ h[6];
    h[1] = h[2] ^ h[5] ^ h[6];
    h[3] = h[4] ^ h[5] ^ h[6];

    printf("Hamming Code: ");
    for (int i = 0; i < 7; i++)
        printf("%d ", h[i]);
}
```

Program 5: CRC

```
#include <stdio.h>

unsigned short crc(unsigned short poly, unsigned short init, char *m) {
    unsigned short c = init;
    for (int i = 0; m[i]; i++) {
        c ^= (m[i] << 8);

        for (int j = 0; j < 8; j++)
            c = (c & 0x8000) ? (c << 1) ^ poly : c << 1;
    }
    return c;
}

int main() {
    char msg[100];

    printf("Enter message: ");
    scanf("%s", msg);

    printf("CRC-12: %03X\n", crc(0x80F, 0x0000, msg) & 0x0FFF);
    printf("CRC-16: %04X\n", crc(0x8005, 0x0000, msg));
    printf("CRC-CCITT: %04X\n", crc(0x1021, 0xFFFF, msg));
}
```

Program 6: Go Back N

```
#include <stdio.h>

int main() {
    int total, win, lost;

    printf("Enter total number of frames to send: ");
    scanf("%d", &total);

    printf("Enter window size: ");
    scanf("%d", &win);

    for (int s = 0; s < total; ) {
        int e = s + win - 1;
        if (e >= total)
```

```
            e = total - 1;

        printf("\nSender: Sending frames ");
        for (int i = s; i <= e; i++)
            printf("%d ", i);

        printf("\nEnter the frame number to be lost (-1 if none): ");
        scanf("%d", &lost);

        if (lost >= s && lost <= e) {
            printf("Receiver: Frame %d lost! Go back and resend from %d\n",
                   lost, lost);
            s = lost;
        } else {
            printf("Receiver: Acknowledged up to frame %d\n", e);
            s = e + 1;
        }
    }

    printf("\nAll frames sent successfully using Go-Back-N!\n");
}
```

Program 7: Selective Repeat

```
#include <stdio.h>

int main() {
    int total, win;

    printf("Enter total number of frames: ");
    scanf("%d", &total);

    printf("Enter window size: ");
    scanf("%d", &win);

    for (int i = 0; i < total; i++) {
        printf("Frame %d sent\n", i);

        if (i % 3 == 0 && i != 0)
            printf("Frame %d lost → Selectively Resent\n", i);
        else
            printf("ACK %d received\n", i);
    }

    printf("All frames sent and acknowledged successfully!\n");
}
```

Program 8: Leaky Bucket

```
#include <stdio.h>

int main() {
    int cap, rate, n, p, b = 0;

    printf("Enter capacity of bucket: ");
    scanf("%d", &cap);

    printf("Enter output rate per cycle: ");
    scanf("%d", &rate);

    printf("Enter number of incoming packets: ");
    scanf("%d", &n);

    for (int i = 1; i <= n; i++) {
        printf("\nPackets arriving in cycle %d: ", i);
        scanf("%d", &p);

        if ((b + p) > cap) {
            printf("Bucket overflow! Dropped: %d\n", b + p - cap);
            b = cap;
        } else {
            b += p;
        }

        printf("Bucket = %d\n", b);

        int out = (b < rate) ? b : rate;
        b -= out;

        printf("Leaked = %d, Remaining = %d\n", out, b);
    }
}
```

Program 9: Dijkstra

```
#include <stdio.h>
#define INF 9999998

int main() {
    int n, src;

    printf("n: ");
    scanf("%d", &n);

    int g[n][n], dist[n], vis[n];

    printf("adj:\n");
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++) {
            scanf("%d", &g[i][j]);

            if (i != j && g[i][j] == 0)
                g[i][j] = INF;
        }

    printf("source (0..%d): ", n - 1);
    scanf("%d", &src);

    for (int i = 0; i < n; i++) {
        dist[i] = g[src][i];
        vis[i] = 0;
    }

    dist[src] = 0;

    for (int k = 0; k < n; k++) {
        int u = -1, best = INF;

        for (int i = 0; i < n; i++)
            if (!vis[i] && dist[i] < best) {
                best = dist[i];
                u = i;
            }

        if (u < 0)
            break;

        vis[u] = 1;

        for (int v = 0; v < n; v++)
            if (dist[u] + g[u][v] < dist[v])
                dist[v] = dist[u] + g[u][v];
    }

    printf("dist from %d: ", src);

    for (int i = 0; i < n; i++)
        printf("%d ", dist[i] >= INF ? 999 : dist[i]);
}
```

Program 10: Distance Vector

```
#include <stdio.h>
#define INF 999

int main() {
    int n, cost[10][10], dist[10][10];

    printf("Enter number of nodes: ");
    scanf("%d", &n);

    printf("Enter cost matrix:\n");

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            scanf("%d", &cost[i][j]);

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            dist[i][j] = cost[i][j];

    for (int k = 0; k < n; k++)
        for (int i = 0; i < n; i++)
            for (int j = 0; j < n; j++)
                if (dist[i][k] + dist[k][j] < dist[i][j])
                    dist[i][j] = dist[i][k] + dist[k][j];

    printf("\nFinal Distance Table:\n");

    for (int i = 0; i < n; i++) {
        printf("Node %d: ", i);

        for (int j = 0; j < n; j++)
            printf("%d ", dist[i][j]);

        printf("\n");
    }

    return 0;
}
```

Program 11: Stop and Wait

```
#include <stdio.h>

void receiver(int f) {
    printf("Receiver: Frame %d received. Sending ACK %d...\n", f, f);
}

int main() {
    int n, c;

    printf("Enter total number of frames to send: ");
    scanf("%d", &n);

    for (int i = 1; i <= n; i++) {
        printf("\nSender: Sending Frame %d\n", i);

        printf("Receiver: Do you want to ACK Frame %d? (1=Yes, 0=No): ", i);
        scanf("%d", &c);

        if (c == 1) {
            receiver(i);
            printf("Sender: ACK %d received.\n", i);
        } else {
            printf("Sender: ACK %d lost. Retransmitting...\n", i);
```

```
                i--;
            }
        }

        printf("\nAll %d frames sent successfully!\n", n);

        return 0;
    }
```