

## IOT NETWORK TECHNOLOGY LAB

1. **Develop a program to compute**
  - i) **The CRC at the transmitter for the given message and generator polynomial**
  - ii) **The Syndrome at the receiver to detect the possible error during the transmission**

**// Algorithm //**  
**TO FIND CRC CODE**

- Enter the generator & message polynomial and their orders.
- Append 'n' zeroes at the end of the message ( $n=\text{order of generator polynomial}$ )
- Generate CRC code for the message by simulating shift registers-
- Simulation of shift registers-
  - ✓ The shift register size is equal to the order of the generator of the polynomial.
  - ✓ Each bit has two values (current and previous).
  - ✓ Two arrays are initialized to zero, one for storing current values and other for previous values.
- Generation of CRC code
  - ✓ The input to the shift register is modified by Ex-ORing the LSB of the generator with MSB of the message.
  - ✓ If the generator bit is '0', the corresponding bits in shift register are just shifted (the previous value is assigned to the current value).
  - ✓ If the generator bit is '1' the corresponding bit in the shift register is modified by Ex-OR ing with the previous MSB of the generator with the previous bit.
  - ✓ The final value of the bits in the current array gives the CRC code.
- Replace appended zeroes with the CRC code and transmit (display the message to be transmitted).

### Syndrome Computation at the Receiver

- Enter the received message.
- Find CRC code as explained above.
- If the CRC code obtained is non-zero, display that the received message is erroneous.
- If the CRC code is zero, display that the received message is without errors.

Illustration with Example

CRC

Let  $G(x) = 1011$        $\deg g \text{ of } g(x) = 3$   
 data:  $M(x) = 1001$        $\deg g \text{ of } M(x) = 3.$

Augmented data     $1001\underset{\substack{\text{---} \\ \text{deg } g \text{ of } G(x)}}{\underbrace{0\ 0\ 0}}$   
 [deg  $g$  of  $G(x)$ ]

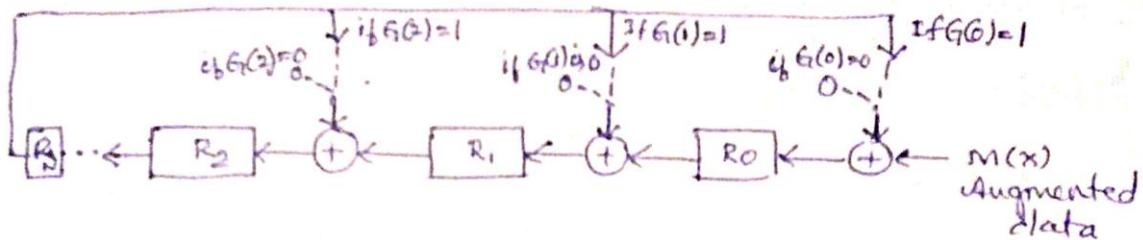
CRC calculation

$$\begin{array}{r}
 \begin{array}{c} 1010 \\ \hline 1011 ) 1001\ 0\ 0\ 0 \end{array} \leftarrow \text{Augmented data.} \\
 \begin{array}{r} 1011 \\ \hline 0100 \\ \hline 0000 \\ \hline 0100 \\ \hline 0100 \\ \hline 1011 \\ \hline 00110 \\ \hline 0000 \\ \hline 0110 \end{array} \text{CRC}
 \end{array}$$

- Points:
- # 1. CRC size is 1 less than length of  $G(x)$
  - # 2. Since CRC is Length of  $\underline{G(x)} - 1$ , we need only those many shift registers for implementation of this algorithm.
  - # 3. Exclusive operations are performed only for "length  $\underline{G(x)} - 1$ " no of bits (indicated by ✓ mark)
  - # 4. NOTICE MSB of Dividend after EX-OR operation is always (✓) zero.
  - # 5. EX-OR operations are needed for only LSB "G(x)-1" no of bits. one of the dividend for EX-OR operation is either "00...0" or LSB  $G(x)$  except "MSB".

## CRC Implementation

### General Structure



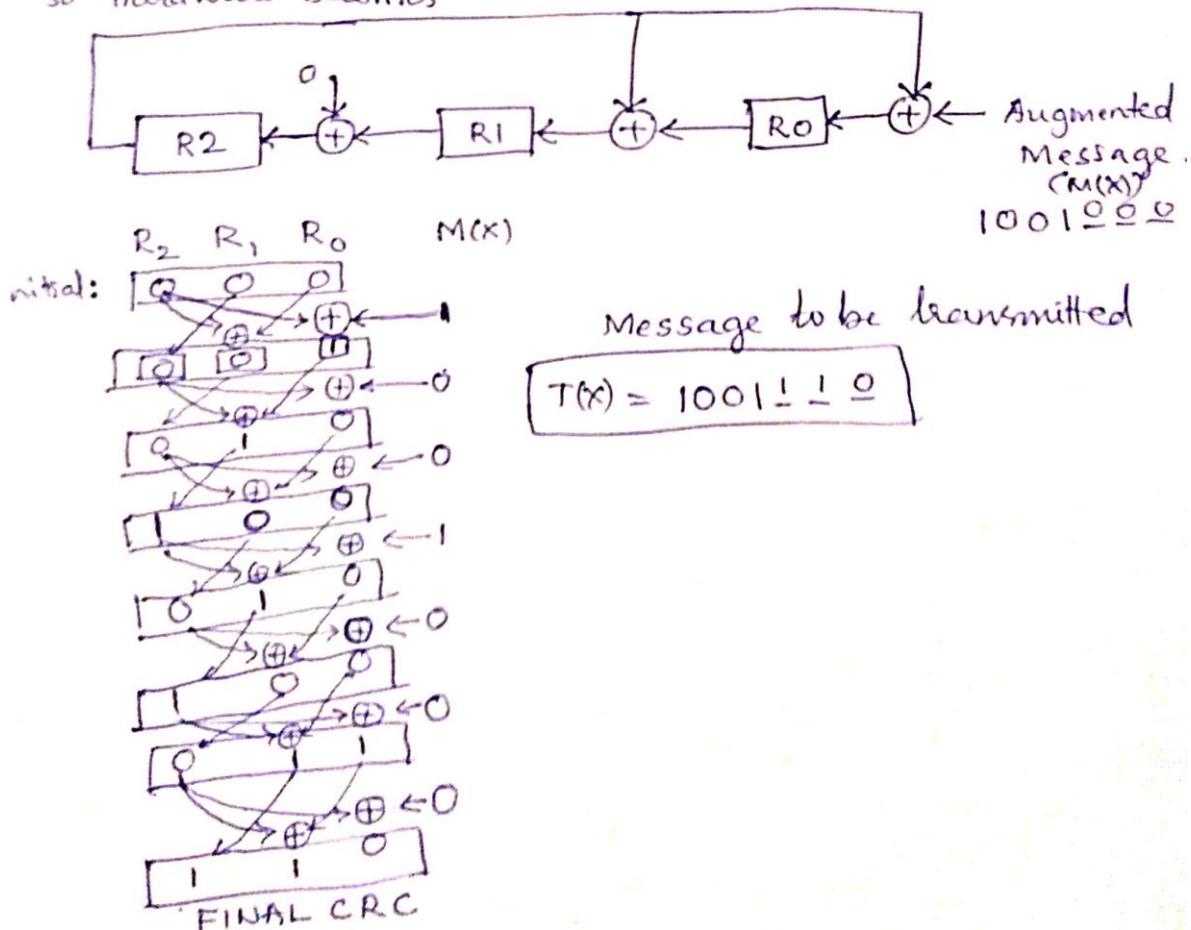
For Given example  $G(X) = 1011$ ,  $M(X) = 1001$ ,

No. of registers required = Length of  $G(X) - 1 = 3$ .

$$\xrightarrow{\text{EX-OR}} \begin{matrix} 1 \\ 1 \end{matrix} = \begin{matrix} 1 \\ 1 \end{matrix} = 3$$

Here  $G(0) = G(1) = 1$     $G(2) = 0$    only these are considered  
 $G(3)$  is not needed.

so Hardware becomes



```

// CODE //

#include<stdio.h>
#include<conio.h>
int dg=16,dm,dt,data[50],gen[17]={1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,1};
int si[16],so[16],tx;

void main()      {

    void crc(int msg[]);
    int i,j,k;

    printf("\n Enter your choice (1/2)\n");
    printf("1 for CRC-CCITT \n");
    printf("2 for generalised generator polynomial\n");
    printf("\n Choice:");

    if(getchar()=='2')      {
        printf("\n Enter the degree of generator polynomial\n");
        scanf("%d",&dg);
        printf("\n Enter the generator polynomial\n");
        for(i=dg;i>=0;i--)      scanf("%d",&gen[i]);
    }

    printf("\n The generator polynomial is \n");
    for(i=dg;i>=0;i--)      printf("%d",gen[i]);

    printf("\n Enter the degree of message\n");
    scanf("%d",&dm);
    printf("\n Enter the message\n");
    for(i=dm;i>=0;i--)      scanf("%d",&data[i]);

    dt=dm+dg;
    for(i=0;i<=dm;i++)      data[dt-i]=data[dm-i];
    for(i=1;i<=dg;i++)      data[dg-i]=0;

    tx=1;
    crc(data);

    printf("\n Enter the received message\n");
    for(i=dt;i>=0;i--)scanf("%d",&data[i]);

    tx=0;
    crc(data);
}

void crc(int msg[])
{
    int i,j,k,flag;

    for(i=0;i<dg;i++)      {
        so[i]=0;
        si[i]=0;
    }
}

```

```

}

for(i=dt;i>=0;i--)      {
    if(gen[0]==1)    si[0]=so[dg-1]^msg[i];
    else si[0]=msg[i];

    for(j=1;j<=dg-1;j++)
        if(gen[j]==1)    si[j]=so[dg-1]^so[j-1];
        else    si[j]=so[j-1];

    printf("\n");
    for(k=dg-1;k>=0;k--)    {
        so[k]=si[k];
        printf("%d",so[k]);
    }
}

if(tx)  {
    printf("\n CRC code is \n");
    for(k=dg-1;k>=0;k--)    {
        printf("%d",so[k]);
        msg[k]=so[k];
    }

    printf("\n Message to be transmitted\n");
    for(i=dt;i>=0;i--)printf("%d",msg[i]);
}

if(!tx)  {
    flag=0;
    for(i=0;i<=dg-1;i++)
        if(so[i]==1)    {
            flag=1;
            break;
        }
    if(flag==0)    printf("\nResult: No error");
    else printf("\n Result: Error in the received msg");
}
}

```

```
Enum in C | Online C Compiler | Enumeration (or e | Bitcoin SV Academy | Online C Compiler | (6) WhatsApp | IoT LAB - Google | + | 
← → C onlinegdb.com/online_c_compiler
Apps IRCTC :: Login Google Live cricket scores... Rediff.com - India... Uttaradi Math - We... Welcome to Facebo... www.kannadaaudio...
|| Other bookmarks | Reading list
[Run] [Debug] [Stop] [Share] [Save] [Beautify] input
Enter your choice (1/2)
1 for CRC-CCTTF
2 for generalised generator polynomial

Choice:2

Enter the degree of generator polynomial
3

Enter the generator polynomial
1 0 1 1

The generator polynomial is
1011
Enter the degree of message
3

Enter the message
1 0 0 1

001
010
100
010
100
011
110
CRC code is
110
Message to be transmitted
1001110
Enter the received message
```

```
Desktop » PM 2:39
23-05-2021
Enum in C | Online C Compiler | Enumeration (or e | Bitcoin SV Academy | Online C Compiler | (6) WhatsApp | IoT LAB - Google | + | 
← → C onlinegdb.com/online_c_compiler
Apps IRCTC :: Login Google Live cricket scores... Rediff.com - India... Uttaradi Math - We... Welcome to Facebo... www.kannadaaudio...
|| Other bookmarks | Reading list
[Run] [Debug] [Stop] [Share] [Save] [Beautify] input
The generator polynomial is
1011
Enter the degree of message
3

Enter the message
1 0 0 1

001
010
100
010
100
011
110
CRC code is
110
Message to be transmitted
1001110
Enter the received message
1 1 0 1 1 1 0

001
011
110
110
110
110
110
Result: Error in the received msg
...Program finished with exit code 35
Press ENTER to exit console.
```

```
Desktop » PM 2:45
23-05-2021
Enum in C | Online C Compiler | Enumeration (or e | Bitcoin SV Academy | Online C Compiler | (6) WhatsApp | IoT LAB - Google | + | 
← → C onlinegdb.com/online_c_compiler
Apps IRCTC :: Login Google Live cricket scores... Rediff.com - India... Uttaradi Math - We... Welcome to Facebo... www.kannadaaudio...
|| Other bookmarks | Reading list
[Run] [Debug] [Stop] [Share] [Save] [Beautify] input
The generator polynomial is
1011
Enter the degree of message
3

Enter the message
1 0 0 1

001
010
100
010
100
011
110
CRC code is
110
Message to be transmitted
1001110
Enter the received message
1 0 0 1 1 1 0

001
010
100
010
101
000
000
Result: No error
...Program finished with exit code 17
Press ENTER to exit console.
```

**2. Develop a program to implement a Shortest path routing algorithm for a given network graph and build a routing table for the given node.**

**SHORTEST PATH ALGORITHM**

1. Enter the distance matrix, which gives the distance of any given node from other nodes. [If the nodes are not directly linked, the distance between them is infinity.]
2. Enter the number of nodes, the source and destination nodes.
3. Create a structure for each node containing details of previous node, length from destination to this node and state label.
4. Start from destination node and find the shortest path to the source node. [The reason for this backward search is that, each node is labeled with its predecessor, while copying the output variable, the path is thus reversed. By reversing the search the two effects cancel and the answer is produced in the correct order.]
  - i. Mark the destination node as permanent node, X.
  - ii. Examine each of the adjacent nodes of the destination node X and the node with the smallest distance becomes the next working node Y.
  - iii. All the adjacent nodes of Y are checked for least distance and the predecessor is changed for the one with shorter path.
  - iv. After all the adjacent nodes have been inspected and modified (if needed), entire graph is searched for the tentatively labeled node with the smallest value.
  - v. The node is made permanent and becomes the working node for the next search.

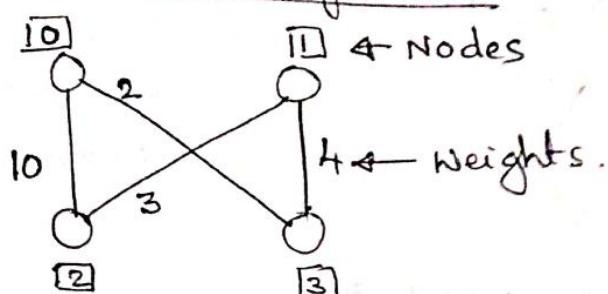
Illustration with an example #1

Shortest path

Dijkstra Algorithm

①

Ex:  
Graph.



# 1 Get

Nodes = 4, Let Source  $[S = 0]$  terminal  $[t = 2]$   
**Algorithm** from user. (destination)

2. Create

$$dist[4][4] = \{ 0, 100, 10, 2 \\ 100, 0, 3, 4 \\ 10, 3, 0, 100 \\ 2, 4, 100, 0 \}$$

3. Create structure for each node with members as follows.

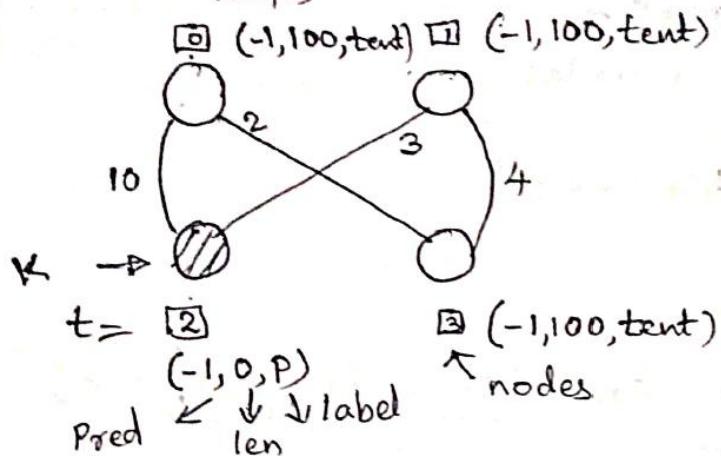
```
state {
    int pred
    int len
    enum (per,temp),Label
}
```

4. Initialize all state node records with  
 $Pred = -1$ ,  $len = 100$ ;  $label = \text{tent}$

5. If  $S = t$ , distance = 0.

6. update the record of destination (termination) node as follows. In this ex: destination =  $t = 2$   
 $\therefore \text{state}[2] = \{ len = 0, label = \text{per} \}$ .

7. So Graph



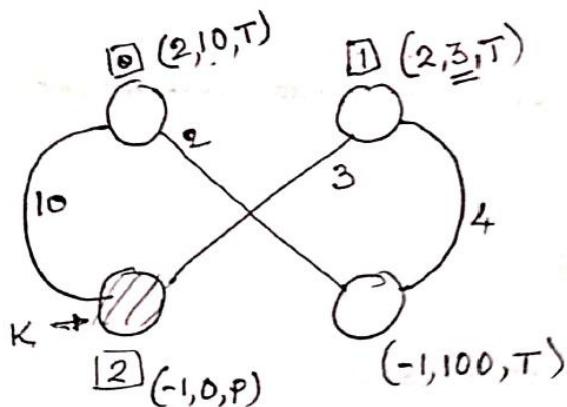
$t = \boxed{2}$

$(-1, 0, P)$

Pred   ↓  
len   ↓ label

8. Make Working node( $k$ ) =  $t \rightarrow$  destination.

- update the attributes (records) of all (including permanent) the nodes in the graph w.r.t. working node.



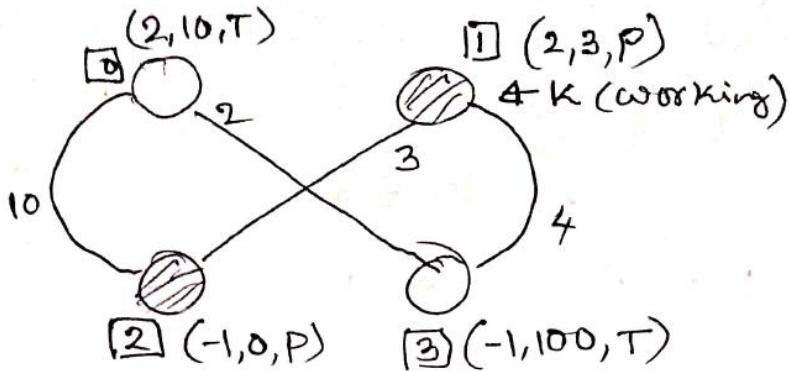
- choose the node whose length is minimum as the working node ' $k$ '

In the above graph node 1 has minimum length(3)  $\therefore k=1$

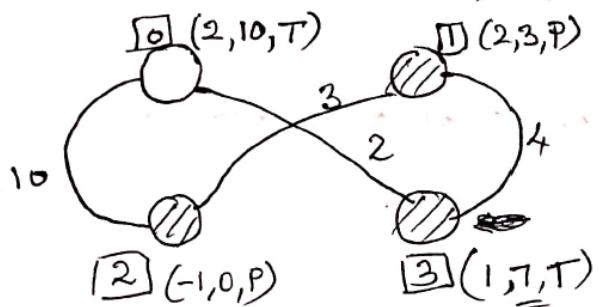
- update node 1 as permanent node.

so graphs

(3)

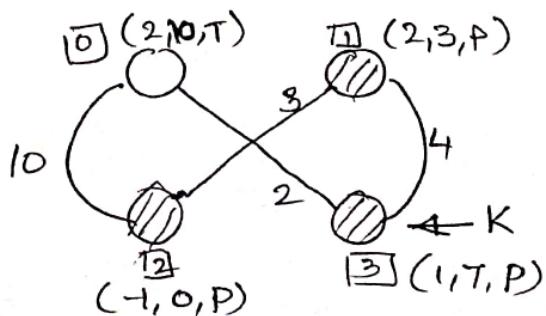


- update the attributes (records) of all tentative nodes considering ~~the working node~~ ~~as~~ node 1, as working node K.



choose the node with minimum length as the next working node  $\Rightarrow K = 3$ . (considering only Tentative nodes)

Make this node as Permanent

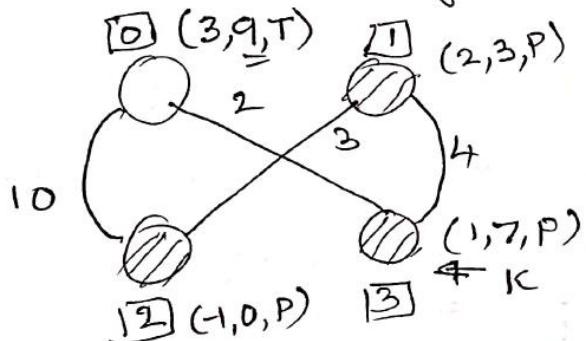


(4)

Repeat the procedure till  $K = \infty$  or working node becomes destination.  
i.e.,  $\underline{K=8}$ . source node

NOW  $K=3$ ,  $S=0$ ,  
Therefore procedure continues.

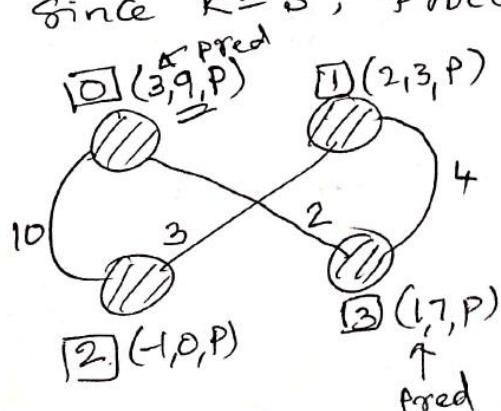
- update all tentative nodes (only node 0 now) considering node 3 as working node  $K$ .



Now find the node with minimum length among the Tentative nodes and make it  $K$ . and permanent.

$\therefore K=0$ , is made permanent.

since  $K=S$ , Procedure stops.



SHORTEST PATH  
Start from source '0'

(S)  $0 \rightarrow 3 \rightarrow 1 + 2$   
Pred Pred Pred (T)

Total dist = Length of  
Source record  
= 9

```

#include<stdio.h>
#include<stdlib.h>
#define nodes 4
#define infinity 100
int n;
int dist[nodes][nodes]=
{0,100,10,2,
100,0,3,4,
10,3,0,100,
2,4,100,0};
void main(){
    int i,s,j,t;
    void shrt();
    printf("\n Enter the no of nodes");
    scanf("%d",&n);
    printf("\n enter the source & dest nodes");
    scanf("%d%d",&s,&t);
    if(s==t){
        printf("\n Source is same as the destination \t Hence the Distance is 0");
        exit(0);
    }
    shrt(s,t);
}
void shrt(int s,int t){
    struct state {
        int predecessor;
        int length;
        enum{per,tent}label;
    }state[nodes];

    int i,k,min;
    struct state *p;
    for(p=&state[0];p<&state[n];p++){
        p->predecessor=-1;
        p->length=infinity;
        p->label=tent;
    }
    state[t].length=0;state[t].label=per;
    k=t;
    do {
        for(i=0;i<n;i++)
            if(dist[k][i]!=0 && state[i].label==tent) {
                if(state[k].length+dist[k][i]<state[i].length) {
                    state[i].predecessor=k;
                    state[i].length=state[k].length+dist[k][i];
                }
            }
        k=0;
        min=infinity;
        for(i=0;i<n;i++)
            if(state[i].label==tent && state[i].length<min) {

```

```

        min=state[i].length;
        k=i;
    }
    state[k].label=per;
}while(k!=s);
k=s;
do {
    printf("%d",k);
    if (k!=t) printf("---->");
    k=state[k].predecessor;
}while(k>=0);
printf("\n The total distance is %d",state[s].length);
}

```

## Results

```

main.c
44         state[i].predecessor=k;
45         state[i].length=state[k].length+dist[k][i];
46     }
47     k=0;
48     min=infinity;
49     for(i=0;i<n;i++)
50         if(state[i].label==tent && state[i].length<min) {
51             min=state[i].length;
52             k=i;
53         }
54     state[k].label=per;
55 }while(k!=s);
56 k=s;
57 do {
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
537
538
539
539
540
541
542
543
544
545
546
547
547
548
549
549
550
551
552
553
554
555
556
557
557
558
559
559
560
561
562
563
564
565
566
566
567
568
568
569
569
570
571
572
573
574
575
575
576
577
577
578
578
579
579
580
581
582
583
584
585
585
586
587
587
588
588
589
589
590
591
592
593
594
595
595
596
597
597
598
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
617
618
619
619
620
620
621
622
623
624
625
626
627
627
628
629
629
630
630
631
632
633
634
635
636
637
637
638
639
639
640
640
641
642
643
644
645
646
646
647
648
648
649
649
650
651
652
653
654
655
656
656
657
658
658
659
659
660
661
662
663
664
665
666
666
667
668
668
669
669
670
671
672
673
674
675
675
676
677
677
678
678
679
679
680
681
682
683
684
685
685
686
687
687
688
688
689
689
690
691
692
693
694
695
695
696
697
697
698
698
699
699
700
701
702
703
704
705
705
706
707
707
708
708
709
709
710
711
712
713
714
715
715
716
717
717
718
718
719
719
720
721
722
723
724
725
725
726
727
727
728
728
729
729
730
731
732
733
734
735
735
736
737
737
738
738
739
739
740
741
742
743
744
745
745
746
747
747
748
748
749
749
750
751
752
753
754
755
755
756
757
757
758
758
759
759
760
761
762
763
764
765
765
766
767
767
768
768
769
769
770
771
772
773
774
775
775
776
777
777
778
778
779
779
780
781
782
783
784
785
785
786
787
787
788
788
789
789
790
791
792
793
794
795
795
796
797
797
798
798
799
799
800
801
802
803
804
805
805
806
807
807
808
808
809
809
810
811
812
813
814
815
815
816
817
817
818
818
819
819
820
821
822
823
824
825
825
826
827
827
828
828
829
829
830
831
832
833
834
835
835
836
837
837
838
838
839
839
840
841
842
843
844
845
845
846
847
847
848
848
849
849
850
851
852
853
854
855
855
856
857
857
858
858
859
859
860
861
862
863
864
865
865
866
867
867
868
868
869
869
870
871
872
873
874
875
875
876
877
877
878
878
879
879
880
881
882
883
884
885
885
886
887
887
888
888
889
889
890
891
892
893
894
895
895
896
897
897
898
898
899
899
900
901
902
903
904
905
905
906
907
907
908
908
909
909
910
911
912
913
914
914
915
916
916
917
917
918
918
919
919
920
921
922
923
924
925
925
926
927
927
928
928
929
929
930
931
932
933
934
935
935
936
937
937
938
938
939
939
940
941
942
943
944
944
945
946
946
947
947
948
948
949
949
950
951
952
953
954
955
955
956
957
957
958
958
959
959
960
961
962
963
964
965
965
966
967
967
968
968
969
969
970
971
972
973
974
975
975
976
977
977
978
978
979
979
980
981
982
983
984
985
985
986
987
987
988
988
989
989
990
991
992
993
994
994
995
996
996
997
998
998
999
999
1000
1001
1002
1003
1004
1005
1005
1006
1007
1007
1008
1008
1009
1009
1010
1011
1012
1013
1014
1014
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1021
1022
1023
1024
1024
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1031
1032
1033
1034
1034
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1041
1042
1043
1044
1044
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1051
1052
1053
1054
1054
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1061
1062
1063
1064
1064
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1071
1072
1073
1074
1074
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1081
1082
1083
1084
1084
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1091
1092
1093
1094
1094
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1101
1102
1103
1104
1104
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1111
1112
1113
1114
1114
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1121
1122
1123
1124
1124
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1131
1132
1133
1134
1134
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1141
1142
1143
1144
1144
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1151
1152
1153
1154
1154
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1161
1162
1163
1164
1164
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1171
1172
1173
1174
1174
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1181
1182
1183
1184
1184
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1191
1192
1193
1194
1194
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1201
1202
1203
1204
1204
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1211
1212
1213
1214
1214
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1221
1222
1223
1224
1224
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1231
1232
1233
1234
1234
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1241
1242
1243
1244
1244
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1251
1252
1253
1254
1254
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1261
1262
1263
1264
1264
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1271
1272
1273
1274
1274
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1281
1282
1283
1284
1284
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1294
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1304
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1311
1312
1313
1314
1314
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1321
1322
1323
1324
1324
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1331
1332
1333
1334
1334
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1341
1342
1343
1344
1344
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1351
1352
1353
1354
1354
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1361
1362
1363
1364
1364
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1371
1372
1373
1374
1374
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1381
1382
1383
1384
1384
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1391
1392
1393
1394
1394
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1401
1402
1403
1404
1404
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1411
1412
1413
1414
1414
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1421
1422
1423
1424
1424
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1431
1432
1433
1434
1434
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1441
1442
1443
1444
1444
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1451
1452
1453
1454
1454
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1461
1462
1463
1464
1464
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1471
1472
1473
1474
1474
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1481
1482
1483
1484
1484
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1491
1492
1493
1494
1494
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1501
1502
1503
1504
1504
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1511
1512
1513
1514
1514
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1521
1522
1523
1524
1524
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1531
1532
1533
1534
1534
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1541
1542
1543
1544
1544
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1551
1552
1553
1554
1554
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1561
1562
1563
1564
1564
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1571
1572
1573
1574
1574
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1581
1582
1583
1584
1584
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1591
1592
1593
1594
1594
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1601
1602
1603
1604
1604
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1611
1612
1613
1614
1614
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1621
1622
1623
1624
1624
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1631
1632
1633
1634
1634
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1641
1642
1643
1644
1644
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1651
1652
1653
1654
1654
1655
1656
1656
1657
1657
1658
1658
1659
1659
1660
1661
1662
1663
1664
1664
1665
1666
1666
1667
1667
1668
1668
1669
1669
1670
1671
1672
1673
1674
1674
1675
1676
1676
1677
1677
1678
1678
1679
1679
1680
1681
1682
1683
1684
1684
1685
1686
1686
1687
1687
1688
1688
1689
1689
1690
1691
1692
1693
1694
1694
1695
1696
1696
1697
1697
1698
1698
1699
1699
1700
1701
1702
1703
1704
1704
1705
1706
1706
1707
1707
1708
1708
1709
1709
1710
1711
1712
1713
1714
1714
1715
1716
1716
1717
1717
1718
1718
1719
1719
1720
1721
1722
1723
1724
1724
1725
1726
1726
1727
1727
1728
1728
1729
1729
1730
1731
1732
1733
1734
1734
1735
1736
1736
1737
1737
1738
1738
1739
1739
1740
1741
1742
1743
1744
1744
1745
1746
1746
1747
1747
1748
1748
1749
1749
1750
1751
1752
1753
1754
1754
1755
1756
1756
1757
1757
1758
1758
1759
1759
1760
1761
1762
1763
1764
1764
1765
1766
1766
1767
1767
1768
1768
1769
1769
1770
1771
1772
1773
1774
1774
1775
1776
1776
1777
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1784
1785
1786
1786
1787
1787
1788
1788
1789
1789
1790
1791
1792
1793
1794
1794
1795
1796
1796
1797
1797
1798
1798
1799
1799
1800
1801
1802
1803
1804
1804
1805
1806
1806
1807
1807
1808
1808
1809
1809
1810
1811
1812
1813
1814
1814
1815
1816
1816
1817
1817
1818
1818
1819
1819
1820
1821
1822
1823
1824
1824
1825
1826
1826
1827
1827
1828
1828
1829
1829
1830
1831
1832
1833
1834
1834
1835
1836
1836
1837
1837
1838
1838
1839
1839
1840
1841
1842
1843
1844
1844
1845
1846
18
```

```

void main(){
    int i,s,j,t;
    void shrt();
    printf("\n Enter the no of nodes");
    scanf("%d",&n);
    printf("\n enter the source & dest nodes");
    scanf("%d%d",&s,&t);
    if(s==t){
        printf("\n Source is same as the destination \t
Hence the Distance is 0");
        exit(0);
    }
    shrt(s,t);
}
void shrt(int s,int t){
    struct state {
        int predecessor;
        int length;
        enum{per,tent}label;
    }state[nodes];

    int i,k,min;
    struct state *p;
    for(p=&state[0];p<&state[n];p++){
        p->predecessor=-1;
        p->length=infinity;
        p->label=tent;
    }
    state[t].length=0;state[t].label=per;
    k=t;
    do {
        for(i=0;i<n;i++)
            if(dist[k][i]!=0 && state[i].label==tent) {
                if(state[k].length+dist[k][i]<state[i].length) {
                    state[i].predecessor=k;
                    state[i].length=state[k].length+dist[k][i];
                }
            }
        k=0;
        min=infinity;
        for(i=0;i<n;i++)
            if(state[i].label==tent && state[i].length<min) {
                min=state[i].length;
                k=i;
            }
        state[k].label=per;
    }while(k!=s);
    k=s;
    do {
        printf("%d",k);
        if (k!=t) printf("->");
    }
}

```

	0	1	2	3	4	5	6	7
0	0	2	1					
1	2	0	2		9			
2		2	0	7		2		
3	1		7	0			7	
4		9			0	4		6
5			2		4	0	2	
6				7	2	0	2	
7					6	2	0	

```

    k=state[k].predecessor;
}while(k>=0);
printf("\n The total distance is %d",state[s].length);
}

```

```

main.c
49         state[i].length=state[k].length+dist[k][i];
50     }
51     k=0;
52     min=infinity;
53     for(i=0;i<n;i++)
54     if(state[i].label==tent && state[i].length<min) {
55         min=state[i].length;
56         k=i;
57     }
58     state[k].label=per;
59 }while(k!=s);
60 k=s;
61
input
Enter the no of nodes 8
enter the source & dest nodes 0 7
0--->1--->2--->5--->6--->7
The total distance is 10
...Program finished with exit code 26
Press ENTER to exit console.

```

```

main.c
49         state[i].length=state[k].length+dist[k][i];
50     }
51     k=0;
52     min=infinity;
53     for(i=0;i<n;i++)
54     if(state[i].label==tent && state[i].length<min) {
55         min=state[i].length;
56         k=i;
57     }
58     state[k].label=per;
59 }while(k!=s);
60 k=s;
61
input
Enter the no of nodes 8
enter the source & dest nodes 7 0
7--->6--->3--->0
The total distance is 10
...Program finished with exit code 26
Press ENTER to exit console.

```

The screenshot shows a web-based C compiler interface. At the top, there are several tabs open in a browser, including "Enum in C", "Online C Compiler - On", "Enumeration (or enum)", "Bitcoin SV Academy | B", "Online C Compiler - on", and "(5) WhatsApp". Below the tabs, the browser's address bar shows "onlinegdb.com/online\_c\_compiler". The main content area is a code editor with a toolbar above it containing buttons for Run, Debug, Stop, Share, Save, and Beautify. The code editor displays a file named "main.c" with the following C code:

```
49     state[i].length=state[k].length+dist[k][i];
50 }
51 k=0;
52 min=infinity;
53 for(i=0;i<n;i++)
54     if(state[i].label==tent && state[i].length<min) {
55         min=state[i].length;
56         k=i;
57     }
58 state[k].label=per;
59 }while(k!=s);
60 k=s;
```

Below the code editor is a terminal window showing the output of the program. The user enters "Enter the no of nodes 8" followed by "enter the source & dest nodes 3 0". The program responds with "3-->0" and "The total distance is 1". The terminal concludes with "...Program finished with exit code 25" and "Press ENTER to exit console.".

This screenshot shows a second execution of the same C program on the same web-based compiler. The setup is identical to the first one, with tabs for "Enum in C", "Online C Compiler - On", "Enumeration (or enum)", "Bitcoin SV Academy | B", "Online C Compiler - on", and "(5) WhatsApp" at the top. The browser address is "onlinegdb.com/online\_c\_compiler".

The code editor in the main content area shows the same "main.c" file with the same C code as the first screenshot.

The terminal window below shows the user entering "Enter the no of nodes 8" and "enter the source & dest nodes 5 7". The program responds with "5-->6-->7" and "The total distance is 4". The terminal concludes with "...Program finished with exit code 25" and "Press ENTER to exit console.".

The screenshot shows a web-based C compiler interface. The code in main.c is as follows:

```
49         state[i].length=state[k].length+dist[k][i];
50     }
51     k=0;
52     min=infinity;
53     for(i=0;i<n;i++)
54     {
55         if(state[i].label==tent && state[i].length<min) {
56             min=state[i].length;
57             k=i;
58         }
59         state[k].label=per;
60     }while(k!=s);
61     k=s;
```

The input window contains the following command-line interaction:

```
Enter the no of nodes 8
enter the source & dest nodes 6 1
6--->5--->2--->1
The total distance is 6
...Program finished with exit code 25
Press ENTER to exit console.
```

The screenshot shows a web-based C compiler interface. The code in main.c is identical to the one in the previous screenshot:

```
49         state[i].length=state[k].length+dist[k][i];
50     }
51     k=0;
52     min=infinity;
53     for(i=0;i<n;i++)
54     {
55         if(state[i].label==tent && state[i].length<min) {
56             min=state[i].length;
57             k=i;
58         }
59         state[k].label=per;
60     }while(k!=s);
61     k=s;
```

The input window contains the following command-line interaction:

```
Enter the no of nodes 8
enter the source & dest nodes 2 2
Source is same as the destination      Hence the Distance is 0
...Program finished with exit code 0
Press ENTER to exit console.
```

### **3. Develop a C program to implement Bit stuffing and De-stuffing using HDLC standard**

#### **Algorithm:**

- Enter the message.
- Check for 5 consecutive ones, if so, append a '0'.
- Append header and trailer flags (both are "0111110") and display the entire frame.
- Accept the frame.
- Sliding through the entered bit stream identify header and trailer flags.
- If flags are not found display suitable message and do not de-stuff.
- If flags are found retrieve the message.
- De-stuff as follows
  - Check for 5 consecutive ones.
  - If so skip a bit (zero appended before) after 5 consecutive ones.
- Display the de-stuffed message.

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

int main()
{
    int i,j=0,count=0,temp=0,beg_msg=0,end_msg=0;
    char msg[100],smsg[100],bsmsg[100],rmsg[100],omsg[100];
    printf("\n Enter the message:\n");
    scanf("%s",msg);

    for(i=0,j=0;msg[i]!='\0';i++,j++){
        smsg[j]=msg[i];
        if(msg[i]=='1')                                /* count the no of successive ones */
            count++;
        else
            count=0;
        if(count==5){                                /* stuff a zero after 5 ones */
            j++;
            smsg[j]='0';
            count=0;
        }
    }
    smsg[j]='\0';
    printf("\n The stuffed message :\n %s",smsg);
    strcpy(bsmsg,"0111110");           /* add header bits */
    strcat(bsmsg,smsg);
    strcat(bsmsg,"0111110");           /* add the tailor bits*/
    printf("\n The frame to be transmitted :\n %s",bsmsg);

    /* destuffing */

    printf("\n Enter the received message :\n");
    scanf("%s",rmsg);
    temp=0;
    i=j=0;
    for(i=0;rmsg[i+7]!='\0';i++){
        if(rmsg[i]=='0' && rmsg[i+7]=='0'){          /* check for header & tailor*/
            for(j=1;j<7;j++){
                if(rmsg[i+j]=='1')
                    temp++;
                else {

```

```

        temp=0;
        break;
    }
}

if(temp==6){
    if(beg_msg==0)                                /* if 01111110 bit stream found */
        beg_msg=i+8;                            /* for the first time, msg begins */
    else{
        end_msg=i;                           /* for the second time msg ends */
        break;
    }
}
else
temp=0;
}

if((beg_msg==0) || (end_msg==0)){
    printf("\n framing error");
    exit(0);
}
count=0;

for(i=beg_msg,j=0;i<end_msg;i++,j++){
    omsg[j]=rmsg[i];
    if(rmsg[i]=='1')
        count++;                         /* check for consecutive ones */
    else
        count=0;                         /* if 0 is found after 5 ones */
    if(count==5){
        i++;                            /* ignore it */
        count=0;
    }
}
omsg[j]='\0';
printf("\n The original message :\n %s",omsg);
return 0;
}

```

```

main.c
1 #include<stdio.h>
2 #include<string.h>
3 #include<stdlib.h>
4
5 int main()
6 {
7     int i,j=0,count=0,beg_msg=0,end_msg=0;
8     char msg[100],smse[100],bsmse[100],rmsse[100],omse[100];
9
10    Enter the message:
11111101001
12
13    The stuffed message :
14    1111111011111010010111110
15    The frame to be transmitted :
16    0111111011111010010111110
17    Enter the received message :
18    0111111011111010010111110
19
20    The original message :
21    111111001
22
23    ...Program finished with exit code 0
24 Press ENTER to exit console.

```

```

main.c
1 #include<stdio.h>
2 #include<string.h>
3 #include<stdlib.h>
4
5 int main()
6 {
7     int i,j=0,count=0,temp=0,beg_msg=0,end_msg=0;
8     char msg[100],smsg[100],bsmsg[100],rmsg[100],omsg[100];

```

Enter the message:  
0111110

The stuffed message :  
011111010  
The frame to be transmitted :  
0111110011110100111110  
Enter the received message :  
011011101111100011111  
framing error

...Program finished with exit code 0  
Press ENTER to exit console.

#### 4. Develop a C program to implement Character stuffing and De-stuffing using HDLC standard

- Enter the message.
- Check for “DLE” in the entire message.
- If found, append another “DLE”.
- Append header (DLE STX) and footer (DLEETX) and display the frame to be transmitted.
- Enter the received message.
- Sliding through the characters entered identify the header and footer.
- If found destuff as follows.
  - Remove the header and footer.
  - In the message check for two consecutive “DLE”.
  - If so skip one “DLE”(one appended before).
- If not found display suitable message and do not destuff.
- Display the destuffed message.

#### Character Stuffing Program

```

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

void main()
{
    int i,j=0,hfound=0,tfound=0,error=0;
    char msg[100],smsg[100],tmsg[100],rmsg[100],dmsg[100];

    printf("\nEnter the message:\n");
    scanf("%s",msg);

    *(smsg)='\0';
    *(dmsg)='\0';

```

```

for(i=0,j=0;msg[i]!='\0';i++,j++)
{
    if(!strncmp(msg+i,"DLE",3))
    {
        smsg[j]='\0';
        strcat(smsg,"DLEDLE");
        i+=2;
        j+=5;
    }
    else smsg[j]=msg[i];
}
smsg[j]='\0';

strcpy(tmsg,"DLESTX");
strcat(tmsg,smsg);
strcat(tmsg,"DLEETX");

printf("\nThe stuffed message is :\n%s",tmsg);

/* Destuffing */

printf("\nEnter the received message :\n");
scanf("%s",rmsg);

for(i=0,j=0;rmsg[i]!='\0';i++)
{
    if(!strncmp(rmsg+i,"DLE",3))
    {
        i+=3;
        if(!strncmp(rmsg+i,"STX",3))
        {
            if(!hfound)      hfound=1;
            else           error=1;
            i+=2;
            continue;
        }
        else if(!strncmp(rmsg+i,"ETX",3))
        {
            tfound=1;
            break;
        }
        else if(!strncmp(rmsg+i,"DLE",3));
        else error=1;
    }
    if(hfound==1)
    {
        dmsg[j]=rmsg[i];
        dmsg[j+1]='\0';
        j++;
    }
}
dmsg[j]='\0';

if(error || !hfound || !tfound)
{
    printf("\nFraming Error\n ");
    exit(0);
}
printf("\nThe destuffed message is :\n%s",dmsg);

}

```

The screenshot shows a web browser window with multiple tabs open. The active tab is "Online C Compiler". The code in the editor is:

```
main.c
56     else if(!strncmp(rmsg+i,"DLE",3));
57     else error=1;
58
59     if(hfound==1)
60     {
61         dmsg[j]=rmsg[i];
62         dmsg[j+1]='\0';
63     }

```

The output window shows the following interaction:

```
Enter the message:
DLEDLEHAIHOWAREYOU

The stuffed message is :
DLESTXLDLEDLEHAIHOWAREYODLEETX
Enter the received message :
DLESTXLDLEDLEHAIHOWAREYODLEETX

The destuffed message is :
DLEDLEHAIHOWAREYOU

...Program finished with exit code 0
Press ENTER to exit console.
```

The screenshot shows a web browser window with multiple tabs open. The active tab is "Online C Compiler". The code in the editor is identical to the first screenshot.

The output window shows the following interaction:

```
Enter the message:
DLEETX

The stuffed message is :
DLESTXLDLEDLEETXDLEETX
Enter the received message :
DLESTXLDLEDLEETXETX

Framing Error

...Program finished with exit code 0
Press ENTER to exit console.
```

The screenshot shows a web browser window with multiple tabs open. The active tab is "Online C Compiler". The code in the editor is identical to the previous screenshots.

The output window shows the following interaction:

```
Enter the message:
ETXDLE

The stuffed message is :
DLESTXETXDLEDLEETTX
Enter the received message :
DLESTXETXDLEDLEETTX

The destuffed message is :
ETXOLE

...Program finished with exit code 0
Press ENTER to exit console.
```

## 5. Develop a C program to implement Encryption by Transposition algorithm

Encryption by transposition method

Plaintext: message

Cipher text: Encrypted message

Ex: plaintext: indiaisfighting covid  
Key: doctor

# Step 1: Prepare text matrix.

	1	2	3	4	5
d	o	c	t	o	r
i	n	d	i	a	i
s	t	i	g	h	t
i	n	g	c	o	n
i	d	a	b	c	d

Prepare the text matrix by arranging rowwise

#2 Chronological order of the key: c d o o r t  
[2, 0, 1, 4, 5, 3]

#3. Encrypted matrix

c	d	o	o	r	t
d	i	n	a	i	i
i	s	f	h	t	g
g	i	n	o	v	c
a	i	d	c	d	b

Ciphertext

dig*a*isi*in*fd*nd*  
ah*o*c*itv*dig*cb*  
(Display the ciphertext columnwise)

# 4 At the Receiver if Key is Same...

Decrypt the message

d	o	c	t	o	r
i	n	d	i	a	i
s	t	i	g	h	t
i	n	g	c	o	v
i	d	a	b	c	d

Display row wise

Decrypted message  
IndiaisfightingCovid

- Get the data and keyword from user.
- Calculate the length of data and keyword.
- Obtain order matrix in which the position of each letter in the key word is stored, by alphabetical arrangement
- Arrange the plain text in the form of a matrix (encryption matrix) as follows-
  - Number of columns is equal to the keyword length.
  - Divide the message length by keyword length.
  - If remainder is non-zero, number of rows is one more than the quotient.
  - Copy the plain text into the encryption matrix row wise.
  - Append redundant characters in unfilled columns (a, b...)
- The encrypted message is obtained by reading text column wise from the encrypted matrix, whose order is specified in the order matrix
- Decryption
- Get keyword and length of message from user.
- Compare with the original keyword, if not matched, display a message and continue if requested.
- If matched decrypt the received message by reading it row wise, neglecting the redundant characters.

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
    char txt[50],kw[10],kw1[10],temp,txt1[10][10],encr[10][10],decr[10][10];
    int lenm,lenk,order[10],i,j,k,temp1,c=0,r,b,count;

    printf("\nEnter the message to be encrypted: ");
    scanf("%s",txt);
    lenm=strlen(txt);

    printf("\nEnter the keyword: ");
    scanf("%s",kw);
    lenk=strlen(kw);

    strcpy(kw1,kw);

    for(i=0;i<lenk;i++)
        order[i]=i;

    //sort the keyword
    for(i=0;i<lenk;i++)
        for(j=0;j<lenk;j++) {
            if(kw1[j]>kw1[i]) {
                temp=kw1[j];
                kw1[j]=kw1[i];
                kw1[i]=temp;

                temp1=order[j];
                order[j]=order[i];
                order[i]=temp1;
            }
        }
    printf("\n Order of letters after sorting: ");
    for(i=0;i<lenk;i++)
        printf("%d ",order[i]);

    r=lenm/lenk;      //no. of rows
```

```

b=lenm%lenk;

if(b!=0) r++;

//convert message to matrix
count=0;
for(i=0;i<r;i++) {
    for(j=0;j<lenk && count<=lenm;j++)
        txt1[i][j]=txt[count++];
    if(count>lenm) {
        while(b!=lenk) {
            txt1[i][b]='a'+c++;
            b++;
        }
    }
}

//encryption
printf("\nThe encrypted message :\n");
for(k=0;k<lenk;k++) {
    for(i=0;i<r;i++) {
        j=order[k];
        encr[i][k]=txt1[i][j];
        printf("%c",encr[i][k]);
    }
}

//at the receiver
printf("\nEnter the keyword: ");
scanf("%s",kw1);

if(strcmp(kw,kw1)) {
    printf("Wrong Key!!");

    exit(0);
}

//decryption

for(i=0;i<lenk;i++) {
    j=order[i];
    for(k=0;k<r;k++)
        decr[k][j]=encr[k][i];
}

printf("\n The original message is: ");
for(i=0;i<r;i++) {
    for(j=0;j<lenk;j++) {
        if(((i*lenk)+j)==lenm){
            break;
        }
        printf("%c",decr[i][j]);
    }
}
}

```

```
main.c
81     for(K=0;K<r;K++)
82         decr[k][j]=encr[k][i];
83     }
84
85     printf("\n The original message is: ");
86     for(i=0;i<r;i++)
87         for(j=0;j<lenk;j++) {
88             printf("%c",decr[i][j]);
89         }
90     printf("\n");
91 }
```

Enter the message to be encrypted: karnatakaishbad  
Enter the keyword: false  
Order of letters after sorting: 1 4 0 2 3  
The encrypted message : asabaiaktsrkanaad  
Enter the keyword: false  
The original message is: karnatakaishbad  
...Program finished with exit code 0  
Press ENTER to exit console.

```
main.c
81     for(K=0;K<r;K++)
82         decr[k][j]=encr[k][i];
83     }
84
85     printf("\n The original message is: ");
86     for(i=0;i<r;i++)
87         for(j=0;j<lenk;j++) {
88             printf("%c",decr[i][j]);
89         }
90     printf("\n");
91 }
```

Enter the message to be encrypted: todayismonday  
Enter the keyword: day  
Order of letters after sorting: 1 0 2  
The encrypted message : eyndataanydiolah  
Enter the keyword: day  
The original message is: todayismonday  
...Program finished with exit code 0  
Press ENTER to exit console.

The screenshot shows a web-based C compiler interface. The code in the main.c file is:

```
main.c
17:     for(i=0;i<r;i++)
18:         j=order[i];
19:         for(k=0;k<r;k++)
20:             decr[k][j]=encr[k][i];
```

The terminal window shows the following interaction:

```
Enter the message to be encrypted: Indiaisfightingcovid
Enter the keyword: doctor
Order of letters after sorting: 2 0 1 4 5 3
The encrypted message :
dgaisiainfdahocitydigob
Enter the keyword: doctor
The original message is: Indiaisfightingcovid
...Program finished with exit code 0
Press ENTER to exit console.
```

## 6. Develop a C program to implement Encryption by Substitution algorithm

### Algorithm:

- Get the message.
- Replace each character in the plain text by another character at an offset ‘KEY\_SHIFT(3)’ such that the encrypted message has characters in the specified range (English alphabets both upper and lower case in circular method).
- Transmit the message (display the encrypted message).
- Receive the encrypted message.
- Replace each character in the received text by removing the offset ‘KEY\_SHIFT(3)’.
- Display the decrypted message.

/\* CODE \*/

```
#include<stdio.h>
#include<string.h>
void main()
{
    char msg[100],encr[100],decr[100],rec[100];
```

```

int i;

printf("\n Enter the message for encryption:\n ");
scanf("%s",msg);

for(i=0;*(msg+i)!='\0';i++) {
    if((*(msg+i)>='a' && *(msg+i)<='w') || (*(msg+i)>='A' && *(msg+i)<='W'))
        *(enqr+i)=*(msg+i)+3;
    else
        if((*(msg+i)>='x' && *(msg+i)<='z') || (*(msg+i)>='X' && *(msg+i)<='Z'))
            *(enqr+i)=*(msg+i)+3-26;
        else
            *(enqr+i)=*(msg+i)+3;
}
*(enqr+i)='\0';

printf("\n Encrypted message: %s\n",enqr);

/* Decryption */

printf("\n Enter the message for decryption:\n ");
scanf("%s",rec);

for(i=0;*(rec+i)!='\0';i++) {
    if((*(rec+i)>='d' && *(rec+i)<='z') || (*(rec+i)>='D' && *(rec+i)<='Z'))
        *(decr+i)=*(rec+i)-3;
    else
        if((*(rec+i)>='a' && *(rec+i)<='c') || (*(rec+i)>='A' && *(rec+i)<='C'))
            *(decr+i)=*(rec+i)-3+26;
        else
            *(decr+i)=*(rec+i)-3;
}
*(decr+i)='\0';

printf("\n The decrypted message:\n %s",decr);
}

```

The screenshot shows the OnlineGDB IDE interface. The code editor window displays the provided C program. The terminal window below shows the execution process:

```

$ Online C Compiler - online editor x +
main.c
1 #include<stdio.h>
2 #include<string.h>
3
4 void main()
5 {
6     char msg[100],enqr[100],decr[100],rec[100];
7     int i;
8
9     printf("\n Enter the message for encryption:\n ");
10    scanf("%s",msg);
11
Enter the message for encryption:
hailsl19EC999!
Encrypted message: kdl4vl4<HP<<<$

Enter the message for decryption:
kdl4vl4<HP<<<$

The decrypted message:
hailsl19EC999!

...Program finished with exit code 0
Press ENTER to exit console.

```

The terminal output shows the encryption of the message "hailsl19EC999!" to "kdl4vl4<HP<<<\$" and the successful decryption back to the original message.

Online C Compiler - online editor

onlinegdb.com/online\_c\_compiler

Language: C

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

void main()
{
    char msg[100],encr[100],decr[100],rec[100];
    int i;

    printf("\n Enter the message for encryption:\n ");
    scanf("%s",msg);

    Enter the message for encryption:
123abcwxyz#
```

Encrypted message: 456defzabc&

```
Enter the message for decryption:
456defzabc&
```

The decrypted message:
123abcwxyz#

```
...Program finished with exit code 0
Press ENTER to exit console.
```

About • FAQ • Blog • Terms of Use • Contact Us  
• GDB Tutorial • Credits • Privacy

https://www.onlinegdb.com/online\_c\_compiler#tab-stdin

AM 11:45  
20-05-2021

Online C Compiler - online editor

onlinegdb.com/online\_c\_compiler

Language: C

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

void main()
{
    char msg[100],encr[100],decr[100],rec[100];
    int i;

    printf("\n Enter the message for encryption:\n ");
    scanf("%s",msg);

    Enter the message for encryption:
123abcwxyz#
```

Encrypted message: 456defzabc&

```
Enter the message for decryption:
456defzabc&
```

The decrypted message:
123abcwxyz#

```
...Program finished with exit code 0
Press ENTER to exit console.
```

About • FAQ • Blog • Terms of Use • Contact Us  
• GDB Tutorial • Credits • Privacy

https://www.onlinegdb.com/online\_c\_compiler#tab-stdin

AM 11:45  
20-05-2021