

```
% % LEMPEL-ZIV DICTIONARY ENCODING TECHNIQUE.
function[] = lempelziv(seq1,type)
% "seq1" is the string of symbols to be encoded.
% "type" specifies the type of data to be used for encoding(binary,ternary,quaternary,ascii etc.)
%%
% Recognizing the type of data.
% "sym" is an array of possible symbols from a source.
if strcmpi(type,'binary')
    type=2;
    sym='01';
elseif strcmpi(type,'ternary')
    type=3;
    sym='012';
elseif strcmpi(type,'quaternary')
    type=4;
    sym='0123';
elseif strcmpi(type,'ascii')
    type=8;
    sym='01';
elseif strcmpi(type,'char')
    type=10;
    sym=['A':'Z','a':'z','!','@','?',';','.',',','/',' ',' ','&'];
    seq=seq1;
end
%%
% Conversion of source symbols according to specified data type.
if type<8
    seq=zeros(1,0);
end
```

```
2 of 6

if (max(seq1)-48) ~=type-1
no_digits=cceil(log(max(seq1)+1)/log(type));
if no_digits==1
    seq=seq1;
else
    for i=1:length(seq1)
        seq=[seq,dec2base(seq1(i),type,no_digits)];
    end
end
else
    seq=seq1;
end
elseif type==8
    seq=zeros(1,0);
    for i=1:length(seq1)
        seq=[seq,dec2bin(seq1(i),8)];
    end
end
%%
% Finding the phrases from the string.
i=1;
j=1;
k=1;
while k<=length(seq)
    phrases(i).phrase(j)=seq(k);
    j=j+1;
    if k>1
        count=0;
    while count~=i-1 && k<length(seq)
```

```
count=0;
for temp=i-1:-1:1
    if strcmp(phrases(i) .phrase ,phrases(temp) .phrase)
        k=k+1;
        phrases(i) .phrase(j)=seq(k);
        j=j+1;
        break;
    else
        count=count+1;
    end
end
if k<length(seq)
    j=1;
    k=k+1;
    i=i+1;
else
    break;
end
end
if k<length(seq)
    j=1;
    k=k+1;
    i=i+1;
else
    break;
end
end
%% % Finding the unique phrases.
unique=0;
for i=1:length(phrases)-1
    count=0;
    for j=i+1:length(phrases)
        if ~strcmp(phrases(i) .phrase ,phrases(j) .phrase)
            count=count+1;
        end
    end
end
```

```
4 of 6

    end
end
if count==j-i
    unique=unique+1;
end
unique=unique+1;
%%
% Finding the number of digits required to code the phrases.
sym_dim=length(sym);
codelength=log(unique)/log(sym_dim);
codelength=ceil(codelength);
%%
% Creating the dictionary of phrases with their corresponding codewords.
% Here codeword with all zeros' is reserved and is used as a prefix for
% first codeword.
for i=1:unique
count=0;
if i<unique
    dict(i).location=char(zeros(1,codelength)+sym(1));
    j=i;
    k=codelength;
    while(j~=0)
        dict(i).location(k)=char(sym(1+rem(j,sym_dim)));
        k=k-1;
        j=floor(j/sym_dim);
    end
dict(i).content=phrases(i).phrase;
```

```
phr_len=length(phrases(i).phrase);
for temp=i-1:-1:1
    if strcmp(phrases(i).phrase(1:phr_len-1),phrases(temp).phrase) && phr_len>1
        dict(i).codeword=[dict(temp).location,phrases(i).phrase(phr_len)];
        break;
    else
        count=count+1;
    end
end
if count==i-1
    dict(i).codeword=char([zeros(1,codelength)+sym(1),phrases(i).phrase]);
end
%
% Printing Codewords.
str=zeros(1,0);
for i=1:unique
    str=[str,dict(i).codeword];
end
if length(phrases)~=unique
    for i=1:unique
        if strcmp(phrases(length(phrases)).phrase,phrases(i).phrase)
            str=[str,dict(i).codeword];
        end
    end
end
disp('Encoded string is: ');
disp(str);
%
```

```
% Finding the compression ratio.  
comp_ratio=length(str)/length(seq);  
disp('Compression Ratio is:');  
disp(comp_ratio);
```