```
>
> str:="STUFF";
```

$$str := \text{"STUFF"} \quad (1)$$

replace every S with a Q, every T ... etc. Substution cipher.

```
> str;
```

$$\text{"STUFF"} \quad (2)$$

```
> str[1];
```

$$\text{"S"} \quad (3)$$

```
> Alphabet:= "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
```

$$Alphabet := \text{"ABCDEFGHIJKLMNOPQRSTUVWXYZ"} \quad (4)$$

```
> Alphabet[26];
```

$$\text{"Z"} \quad (5)$$

```
> key:="thequickbrownfxjmpdvlazygs";
```

$$key := \text{"thequickbrownfxjmpdvlazygs"} \quad (6)$$

transform STUFF

We can do this with a loop.

```
> with(StringTools):
> SearchText("S",Alphabet);
```

$$19 \quad (7)$$

```
> Alphabet[19];
```

$$\text{"S"} \quad (8)$$

```
> key[19];
```

$$\text{"d"} \quad (9)$$

```
> word:="STUFF";
```

$$word := \text{"STUFF"} \quad (10)$$

```
> length(word);
```

$$5 \quad (11)$$

```
> encrypt:= word->cat(seq(key[SearchText(word[i],Alphabet)],i=1..
  length(word)));
```

$$encrypt := word \rightarrow cat\left(seq\left(key_{SearchText\left(word_i,\ Alphabet\right)},\ i = 1\ ..length(word)\right)\right) \quad (12)$$

```
> decrypt:=word->cat(seq(Alphabet[SearchText(word[i],key)],i=1..
  length(word)));
```

$$decrypt := word \rightarrow cat\left(seq\left(Alphabet_{SearchText\left(word_i,\ key\right)},\ i = 1\ ..length(word)\right)\right) \quad (13)$$

```
> encrypt("YOMAMMA");
```

$$\text{"gxntnnt"} \quad (14)$$

```
> decrypt(%);
```

$$\text{"YOMAMMA"} \quad (15)$$

```
> Caesar:= word->cat(seq(Alphabet[SearchText(word[i],Alphabet)+3],
  i=1..length(word)));
```

(16)

$$Caesar := word \rightarrow cat\left(seq\left(Alphabet_{SearchText\left(word_i,\ Alphabet\right)+3},\ i=1\ ..length(word)\right)\right) \quad (16)$$

```
> Caesar("ABC");
```

"DEF" (17)

```
> Caesar("WXYZ")
```

Warning, inserted missing semicolon at end of statement

"Z" (18)

bad at end of alphabet

```
> 95 mod 26;
```

17 (19)

```
> modp(95,26);
```

17 (20)

```
> Caesar:= word->cat(seq(Alphabet[
    modp(
     SearchText(word[i],Alphabet)-1+3,26)+1
  ],i=1..length(word)));
```

$$Caesar := word \rightarrow cat\left(seq\left(Alphabet_{modp\left(SearchText\left(word_i,\ Alphabet\right)+2,\ 26\right)+1},\ i=1\ ..length(word)\right)\right) \quad (21)$$

```
> Caesar("AXYZ")
```

Warning, inserted missing semicolon at end of statement

"DABC" (22)

```
> Caesar:= proc(word,shift)
   cat(seq(Alphabet[
    modp(
     SearchText(word[i],Alphabet)-1+shift,26)+1
  ],i=1..length(word)));
  end;
```

$$Caesar := \mathbf{proc}(word, shift)\ cat(seq(Alphabet[modp(SearchText(word[i], Alphabet) - 1 + shift, 26) + 1], i=1\ ..length(word)))\ \mathbf{end\ proc} \quad (23)$$

```
> Caesar("SALAD",3);
```

"VDODG" (24)

```
> Caesar(%,-3);
```

"SALAD" (25)

```
> Caesar("Salad",3);
```

"VCCCC" (26)

```
> Julius:= proc(word,shift)
   for i from 1 to length(word) do
    j:=SearchText(word[i],Alphabet);
    if (j=0) then
      error("letter not found in alphabet:",word[i]);
    else
     crypt:=cat(crypt,Alphabet[modp(j-1+shift,26)+1])
    fi;
   od;
   return(crypt);
  end;
```

```
Warning, `i` is implicitly declared local to procedure `Julius`
Warning, `j` is implicitly declared local to procedure `Julius`
Warning, `crypt` is implicitly declared local to procedure
`Julius`
```

```
Julius := proc(word, shift)
    local i, j, crypt;
    for i to length(word) do
        j := SearchText(word[i], Alphabet);
        if j = 0 then
            error "letter not found in alphabet:", word[i]
        else
            crypt := cat(crypt, Alphabet[modp(j - 1 + shift, 26) + 1])
        end if
    end do;
    return crypt
end proc
```

(27)

```
> Julius("Salad",3);
Error, (in Julius) letter not found in alphabet:, a
```

```
> Julius("VENI",4);
```

cryptZIRM (28)

```
> Julius:= proc(word,shift)
    local i,j,crypt;
    global Alphabet;
    crypt:="";
    for i from 1 to length(word) do
      j:=SearchText(word[i],Alphabet);
      if (j=0) then
        error("letter not found in alphabet:",word[i]);
      else
        crypt:=cat(crypt,Alphabet[modp(j-1+shift,26)+1])
      fi;
    od;
    return(crypt);
  end:
```

```
> Julius("VENI",4);
```

"ZIRM" (29)

```
> Julius(%,-4);
```

"VENI" (30)

```
> word;
```

"STUFF" (31)

```
> dumb:=proc()
    word:="glip";
  end:
Warning, `word` is implicitly declared local to procedure `dumb`
```

```
> dumb();
```

"glip" (32)

```
> word;
```

"STUFF" (33)

```
> dumb:=proc()
    global word;
    word:="glip";
  end:
> dumb();
```

"glip" (34)

```
> word;
```

"glip" (35)

```
> Julius:= proc(word,shift)
    local i,j,crypt;
    global Alphabet;
    crypt:="";
    for i from 1 to length(word) do
      j:=SearchText(word[i],Alphabet);
      if (j=0) then
        print("ERROR:letter not found in alphabet:",word[i]);
        crypt:=cat(crypt,"?");
      else
        crypt:=cat(crypt,Alphabet[modp(j-1+shift,26)+1])
      fi;
    od;
    return(crypt);
  end:
> Julius("SalAD",5);
```

"ERROR:letter not found in alphabet:", "a"
"ERROR:letter not found in alphabet:", "l"
"X??Fl" (36)

```
> Ord("A");
```

65 (37)

```
> Char(68);
```

"D" (38)

```
> convert("I have heard the mermaids singing, each to each.",
  bytes);
```

[73, 32, 104, 97, 118, 101, 32, 104, 101, 97, 114, 100, 32, 116, 104, 101, 32, 109, 101, 114, 109, 97, 105, 100, 115, 32, 115, 105, 110, 103, 105, 110, 103, 44, 32, 101, 97, 99, 104, 32, 116, 111, 32, 101, 97, 99, 104, 46] (39)

```
> convert("AaBb",bytes);
```

[65, 97, 66, 98] (40)

```
> ?convert
> convert([1,2,56,42,47],bytes);
```

" 8*/" (41)

```
> convert("\001\0028*/",bytes);
```

[1, 2, 56, 42, 47] (42)

```
> JuliusAscii:= proc(word,shift)
    local i,j,crypt;
    crypt:="";
    for i from 1 to length(word) do
      j:=Ord(word[i]);
      crypt:=cat(crypt,Char(modp(j-1+shift,255)+1))
    od;
```

```
    return(crypt);
  end:
> JuliusAscii("Veni, Vidi, Vici", 18);
                          "hw  {O2h{v{O2h{u{"                    (43)
> JuliusAscii(%,-18);
                            "Veni, Vidi, Vici"                   (44)
```