```
Algo get_list_of_charaters(formula):
{
Returns the characters in the formula as a list
}
Algo put_open_paranthesis(formula):
{
Initialize count=0
Iterate i in reverse direction from length of formula to 1
{
      If formula[i]=)
      {
increase count by 1
      }
If formula[i]=(
{
decrease count by 1
}
If count=0
      {
insert ( at $i^{th}$ position
get out of the loop
}
}
Return formula
}
Algo put_close_parathesis(formula):
{
Initialize count=0
Iterate i from 1 to length of formula
{
If formula[i]=(
{
increase count by 1
}
If formula[i]=)
{
decrease count by 1
}
If count=0
      {
            if formula[i]=!
{
while formula[i] is not an alphabet
{
increase i by 1
}
}
if i is not the last index of formula
{
insert ( at $i^{th}$ position
}
otherwise
{
insert ( at last position
}
}
}
Return formula
}
Algo check_wellform(formula):
{
Iterate I from 1 to length of formula
{
If formula[1]=)
{
retrun false
}
}
Symbol:=get_list_of_charaters("&|>~(")
Symbol1:=get_list_of_charaters("&|>~)")
Iterate i from 2 to length of formula
{
If formula[i]=( and formula[i-1] count in symbol=0
{
return false
}
}
Iterate i from 1 to length-1 of formula
```

```
{
If formula[i]=) and formula[i+1] count in symbol1=0
{
return false
}
}
Otherwise return true
}
```

Algo perform_not_operation(formula):

```
{
If check_wellform(formula) is false
{
Return none
}
If last element in formula is not !
{
Iterate I from 1 to length of formula
{
      If formula[i]=!
{
if formula[i+1] is not anything of (&,|,>,~)
{
insert formula[i] at end of New_formula
increase i by 1
New_formula:=put_open_paranthesis(New_Formula)
put_close_paranthesis(sub-list of formula(i,length of formula))
add the above at end of New_formula
increase i by length of sub-list
if i=length of formula
{
get out of loop
}
}
otherwise return none
}
Otherwise
            insert formula[i] at end of New_formula
            Increase I by 1

}
Return New_formula
}
Otherwise
Return none
}
```

Algo perform_and_operation(formula):

```
{
If check_wellform(formula) is false
{
Return none
}
If 1st element or last element of formula =&
{
Return none
}
Otherwise
{
If & count in formula=1 and (|,>,~) count in formula=0
{
Return formula
}
Iterate i from 1 to length of formula
{
If formula[i]=&
{
if formula[i+1] is not anything of (!,&,|,>,~)
{
                   New_formula:=put_open_paranthesis(New_Formula)
                   insert formula[i] at end of New_formula
increase i by 1
            put_close_paranthesis(sub-list of formula(i,length of formula))
add the above at end of New_formula
increase i by length of sub-list
if formula[i]=&
{
return none
}
if i=length of formula
{
get out of loop
}
}
otherwise return none
}
Otherwise
            insert formula[i] at end of New_formula
            Increase i by 1
}
Return New_Formula
}
Otherwise return None
}
Algo perform_or_operation(formula):
{
If check_wellform(formula) is false
{
Return none
}
If 1st element or last element of formula =|
{
Return none
}
Otherwise
{
If | count in formula=1 and (>,~) count in formula=0
{
Return formula
}
Iterate i from 1 to length of formula
{
If formula[i]=|
{
if formula[i+1] is not anything of (!,&,|,>,~)
{
                   New_formula:=put_open_paranthesis(New_Formula)
                   insert formula[i] at end of New_formula
increase i by 1
            put_close_paranthesis(sub-list of formula(i,length of formula))
add the above at end of New_formula
increase i by length of sub-list
```

```
if formula[i]=|
{
return none
}
if i=length of formula
{
get out of loop
}
}
otherwise return none
}
Otherwise
             insert formula[i] at end of New_formula
             Increase i by 1
}
Return New_Formula
}
Otherwise return None
}
```

Algo perform_implication_operation(formula):

```
{
If check_wellform(formula) is false
{
Return none
}
If 1st element or last element of formula =>
{
Return none
}
Otherwise
{
If > count in formula=1 and ~ count in formula=0
{
Return formula
}
Iterate i from 1 to length of formula
{
If formula[i]=>
{
if formula[i+1] is not anything of (!,&,|,>,~)
{
                   New_formula:=put_open_paranthesis(New_Formula)
                   insert formula[i] at end of New_formula
increase i by 1
            put_close_paranthesis(sub-list of formula(i,length of formula))
add the above at end of New_formula
increase i by length of sub-list
if formula[i]=>
{
return none
}
if i=length of formula
{
get out of loop
}
}
otherwise return none
}
Otherwise
            insert formula[i] at end of New_formula
            Increase i by 1
}
Return New_Formula
}
Otherwise return None
}
```

Algo perform_bidirectional_operation(formula):

```
{
If check_wellform(formula) is false
{
Return none
}
If 1st element or last element of formula =~
{
Return none
}
Otherwise
{
If ~ count in formula=1
{
```

```
Return formula
}
Iterate i from 1 to length of formula
{
If formula[i]=~
{
if formula[i+1] is not anything of (!,&,|,>,~)
{
                   New_formula:=put_open_paranthesis(New_Formula)
                   insert formula[i] at end of New_formula
increase i by 1
             put_close_paranthesis(sub-list of formula(i,length of formula))
add the above at end of New_formula
increase i by length of sub-list
if formula[i]=~
{
return none
}
if i=length of formula
{
get out of loop
}
}
otherwise return none
}
Otherwise
             insert formula[i] at end of New_formula
             Increase i by 1
}
Return New_Formula
}
Otherwise return None
}
Algo main()
{
formula=user input formula as a string
formula_list=get_list_of_characters(formula)
Iterate i from 1 to 5
{
if i=1
```

```
{
New_formula=perform_not_operation(formula_list)
}
if i=2 and New_formula not equals to none
{
New_formula=perform_not_operation(formula_list)
}
if i=3 and New_formula not equals to none
{
New_formula=perform_not_operation(formula_list)
}
if i=4 and New_formula not equals to none
{
New_formula=perform_not_operation(formula_list)
}
if i=5 and New_formula not equals to none
{
New_formula=perform_not_operation(formula_list)
}
}
      if New_formula equals to none
write “not well formed formula”
otherwise
write New_formula as a string
}
```