****

Name: Anish Vaidya
Roll: 62
Div: A
SRN: 201900160

****

Compiler Design Assignment-6: Implement the following code optimizations on the input 3-address code in the form of quadruples:
a) Common subexpression elimination

b) Constant folding

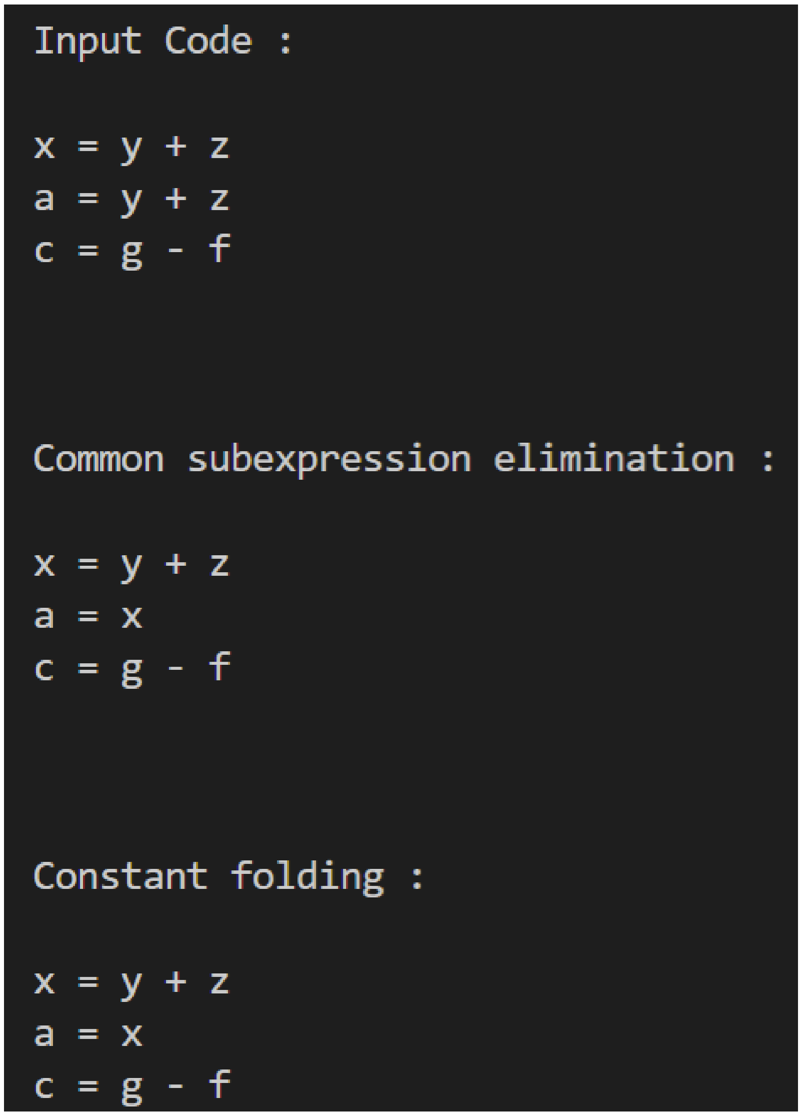
Input :

```
x = y + z

a = y + z

c = g - f
```

OUTPUT:



```
Input Code :

x = y + z
a = y + z
c = g - f



Common subexpression elimination :

x = y + z
a = x
c = g - f



Constant folding :

x = y + z
a = x
c = g - f
```

Code:

```
class Quad():

    def __init__(self,result,arg1,op,arg2):
        self.op = op
        self.arg1 = arg1
        self.arg2 = arg2
        self.result = result
        self.rhs = None
        self.calcRhs()

    def calcRhs(self):
        if(self.arg2 != None):
            self.rhs = f"{self.arg1} {self.op} {self.arg2}"
        else:
            self.rhs = f"{self.arg1}"

IC = []

with open("input.txt","r") as f:
    lines = f.read().split("\n")
    for line in lines:
        comp = line.split()
        if(len(comp) == 3):
            entry = Quad(comp[0],comp[2],"=",None)
        elif(len(comp) == 5):
            entry = Quad(comp[0],comp[2],comp[3],comp[4])
        IC.append(entry)

def cmn_expr():
    for i,stmnt in enumerate(IC):
        value_not_changed = True
        for j in range(0,i):
            if stmnt.rhs == IC[j].rhs:
                for k in range(j,i):
                    if IC[k].result in (stmnt.arg1,stmnt.arg2):
                        value_not_changed = False
                if value_not_changed:
                    stmnt.rhs = IC[j].result
                    stmnt.arg1 = IC[j].result
                    stmnt.op = "="
                    stmnt.arg2 = None

def cnst_fold():
    symbol_table = dict()
    for i,stmnt in enumerate(IC):
        if stmnt.op == "=" and stmnt.arg1.isnumeric() :
            symbol_table[stmnt.result] = stmnt.arg1
            for j in range(i+1,len(IC)):
                if IC[j].arg1 in symbol_table:
                    IC[j].arg1 = symbol_table[IC[j].arg1]
                if IC[j].arg2 in symbol_table:
                    IC[j].arg2 = symbol_table[IC[j].arg2]
                if IC[j].arg2 != None :
                    if IC[j].arg1.isnumeric() and IC[j].arg2.isnumeric() :
                        IC[j].calcRhs()
                        IC[j].rhs = str(eval(IC[j].rhs))
                        IC[j].arg1 = IC[j].rhs
                        IC[j].op = "="
                        IC[j].arg2 = None
                else:
                    IC[j].rhs = IC[j].arg1


print("\nInput Code : \n")
for i in IC:
    print(f"{i.result} = {i.rhs}")
print("\n\n")
```

```
cmn_expr()
print("Common subexpression elimination : \n")
for i in IC:
    print(f"{i.result} = {i.rhs}")
print("\n\n")


cnst_fold()
print("Constant folding : \n")
for i in IC:
    print(f"{i.result} = {i.rhs}")
```