```
#!/usr/bin/python

import re

import math

DATASEGMENT=50

STACKSEGMENT=100

isa={}

datalabel={}

datalist=[]

codelabel={}

codelist=[]


def getfloat(fnum):

    result=0

    sign=int(fnum<0)

    fnum=abs(fnum)

    jump=int(math.floor(math.log(fnum)/math.log(2)))

    twobase=float(2**jump)

    fnum-=twobase

    twobase/=2

    for i in range(0,23):

        print(int(fnum>=twobase),)

        result=(result<<1)+int(fnum>=twobase)

        if (fnum>=twobase):

            fnum-=twobase

        twobase/=2

    print(result)

    result=result | ((jump+127)<<23)

    result=result | (sign<<31)

    return result
```

```
def changemem(str):
    global DATASEGMENT,STACKSEGMENT
    memmode={"SMALL":20,"MEDIUM":50,"LARGE":100,"HUGE":200}
    DATASEGMENT=memmode[str.split(" ")[1]]
    STACKSEGMENT=DATASEGMENT+50
    codelist.append("LHI $29,%d"%(STACKSEGMENT>>16))
    codelist.append("ORI $29,$29,%d"%(STACKSEGMENT&0xffff))
    codelist.append("JMP MAIN")


def strtonum(str):
    n=0
    for c in str:
        n<<=1
        n+=int(c)
    return n


def initial():
    global isa
    finisa=open("isa.txt","r")
    p=re.compile(r'(\w+)\.(\d{6})')
    str=finisa.readline()
    while str:
        splitisa=p.search(str).groups()
        isa[splitisa[0]]=strtonum(splitisa[1])
        str=finisa.readline()
```

```
def getcode(str):
    global isa
    regpos=21
    n=0
    cmd=str.split(" ")[0]
    if cmd=="OUT":
        regpos-=10

    if (False):
        n|=(isa[cmd]<<26)
        if str.count(" ")>0:
            paralist=str.split(" ")[1].split(",")
            for c in paralist:
                if c[0]=='$':
                    n|=(int(c[1:])<<regpos)
                    regpos-=5
                else:
                    temp=int(c)
                    if temp<0:
                        if isa[cmd]>=32 and isa[cmd]<=42:
                            temp+=2**26
                        else:
                            temp+=2**16
                    n=n|temp
        return n

def dataprocess(str):
    global datalabel,datalist
    if len(str)>0:
        datagram=str.split(" ")
        datalabel[datagram[0]]=len(datalist)
        for c in datagram[2].split(","):
            datalist.append(int(c))



def codeprocess(str):
    global codelist,codelabel
    if len(str)>0:
        if str.split(",")[-1] in datalabel.keys():
            str=str.replace(str.split(",")[-1],"%d"%(datalabel[str.split(",")[-1]]+DATASEGMENT))
        codegram=str.split(" ")
        if codegram[0]==".PROC" or codegram[0]==".LABEL":
            codelabel[codegram[1]]=len(codelist)
        elif codegram[0]=="RET":
            codelist.append(str)
            codelist.append("SUB $31,$31,1")
        elif ((codegram[0]=="LDR") and (codegram[1].count(",")==1)):
            targetreg=codegram[1].split(",")[0]
            if codegram[1].split(",")[1].count(".")==0:
                targetvalue=int(codegram[1].split(",")[1])
                if targetvalue<0:
                    targetvalue+=2**32
            else:
                targetvalue=getfloat(float(codegram[1].split(",")[1]))
            codelist.append("LHI %s,%d" %(targetreg,targetvalue>>16))
            codelist.append("ORI %s,%s,%d" %(targetreg,targetreg,targetvalue&0xffff))
        elif codegram[0]=="PUSH":
            codelist.append("STR %s,$29,0"%codegram[1])
            codelist.append("ADDI $29,$29,1")
        elif codegram[0]=="POP":
            codelist.append("SUBI $29,$29,1")
            codelist.append("LDR %s,$29,0"%codegram[1])
        elif codegram[0]=="INT":
            codelist.append("CALL _INT_%s"%codegram[1])
        else:
            codelist.append(str)


def prepro():
    global codelist,datalist
    dataflag=0
    fin=open("procpu.txt","r")
    str=fin.readline()
    while str:
        str=str.rstrip("\r\n")
        #print str
        first=str.split(" ")[0]
        if first==".MODEL":
            changemem(str)
        elif first==".DATA":
            dataflag=1
        elif first==".CODE":
            dataflag=0
        elif dataflag:
            dataprocess(str)
        else:
            codeprocess(str)
        str=fin.readline()
    for elm in codelist:
        if (elm[0]=='J' or elm[0:4]=="CALL") and (elm.split(" ")[1] in codelabel.keys()):
            codelist[codelist.index(elm)]=elm.replace(elm.split(" ")[1],"%d"%(codelabel[elm.split(" ")[1]]-
codelist.index(elm)-1))
    print(datalabel)
    print(datalist)
    print(codelabel)
    print(codelist)
    fin.close()


def outputdata():
    global datalist
    fout=open("ramdata.mif","w")
    fout.write("""WIDTH=32;
DEPTH=256;


ADDRESS_RADIX=UNS;
DATA_RADIX=HEX;


CONTENT BEGIN
""")
    fout.write("[0..%d]:0;\n"%DATASEGMENT)
    for i in range(len(datalist)):
        fout.write("%s:%x;\n"%(i+DATASEGMENT,int(datalist[i])))
    fout.write("[%d..255]:0;\n"%(len(datalist)+DATASEGMENT))
    fout.write("END;\n")
    fout.close()
```

```
def outputcode():
    global codelist
    fout=open("ramcode.mif","w")
    fout.write("""WIDTH=32;
DEPTH=256;

ADDRESS_RADIX=UNS;
DATA_RADIX=HEX;

CONTENT BEGIN
""")
    for i in range(len(codelist)):
        fout.write("%s:%x;\n"%(i,getcode(codelist[i])))
    fout.write("[%d..255]:0;\n"%len(codelist))
    fout.write("END;\n")
    fout.close()

def addint():
    fpro=open("pro.txt","r")
    fint=open("int.txt","r")
    fprocpu=open("procpu.txt","w")
    fprocpu.write(fpro.read())
    fprocpu.write(".LABEL EXIT")
    fprocpu.write("\nJMP EXIT\n\n")
    fprocpu.write(fint.read())
    fpro.close()
    fint.close()
    fprocpu.close()
```

addint()

initial()

prepro()

outputcode()

outputdata()