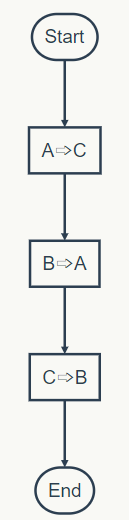
2.

自顶向下实现程序，在向下一层展开前检查本层设计是否有问题，只有上一层是正确的才能向下细化。如果每一层设计都没有问题，整个算法就是正确的。

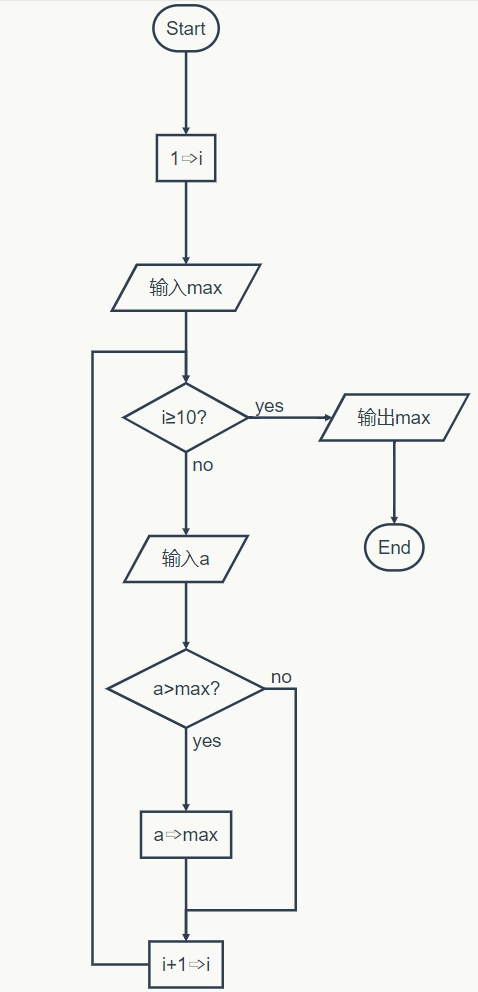
这样每一层向下细化时都不太复杂，容易保证整个算法的正确性。检查时也是从上而下逐层检查，这样做思路清晰，有条不紊地一步一步进行，既严谨又方便。

4.

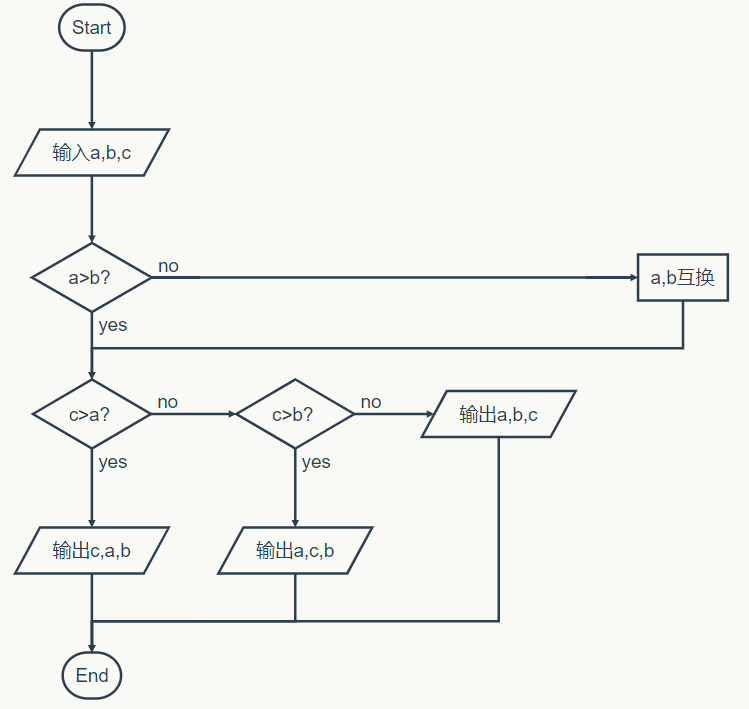
（1）



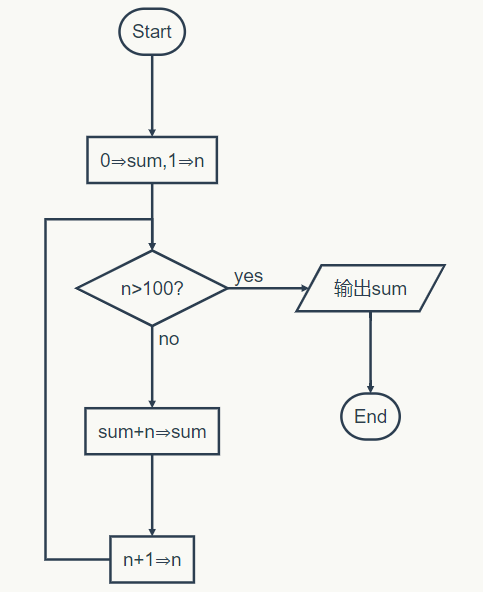
（2）



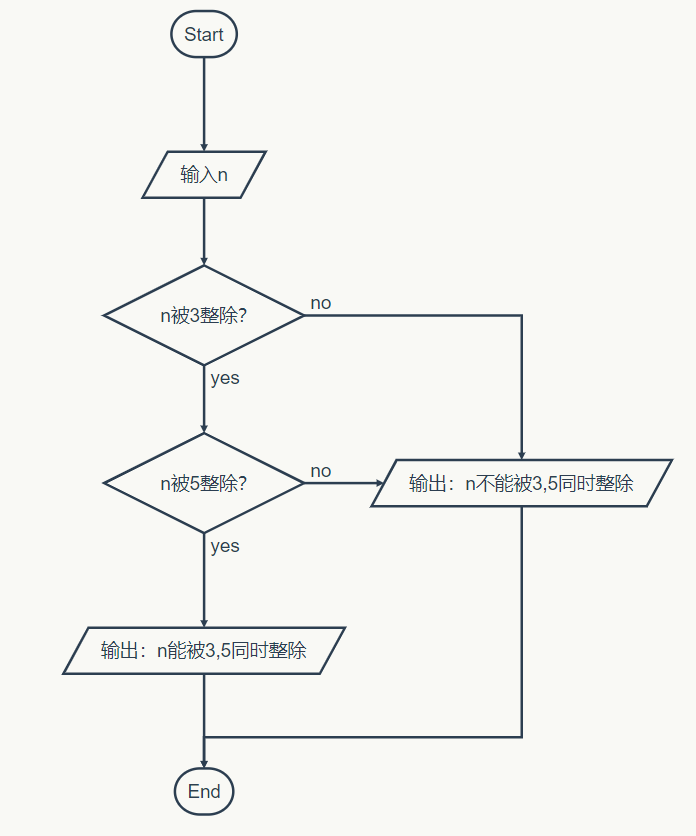
（3）



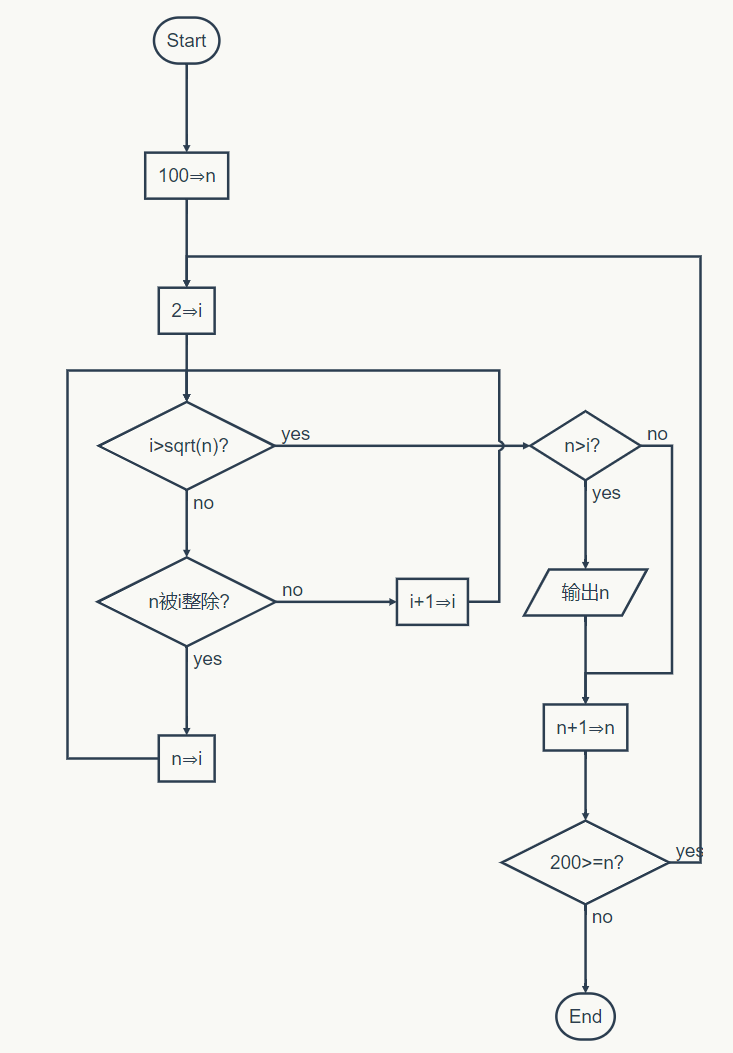
（4）



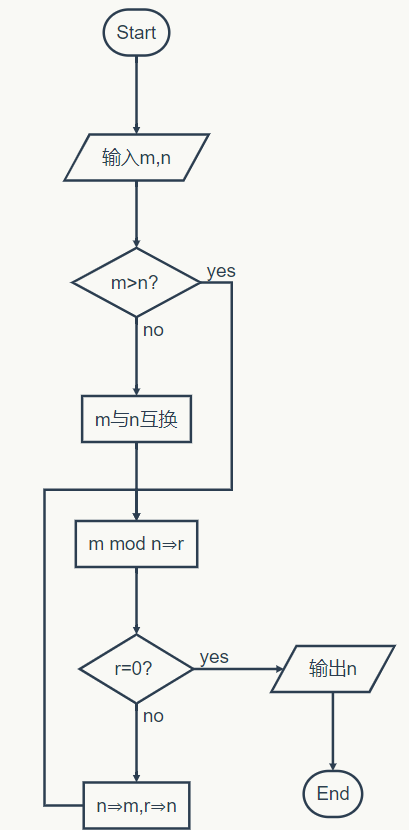
（5）



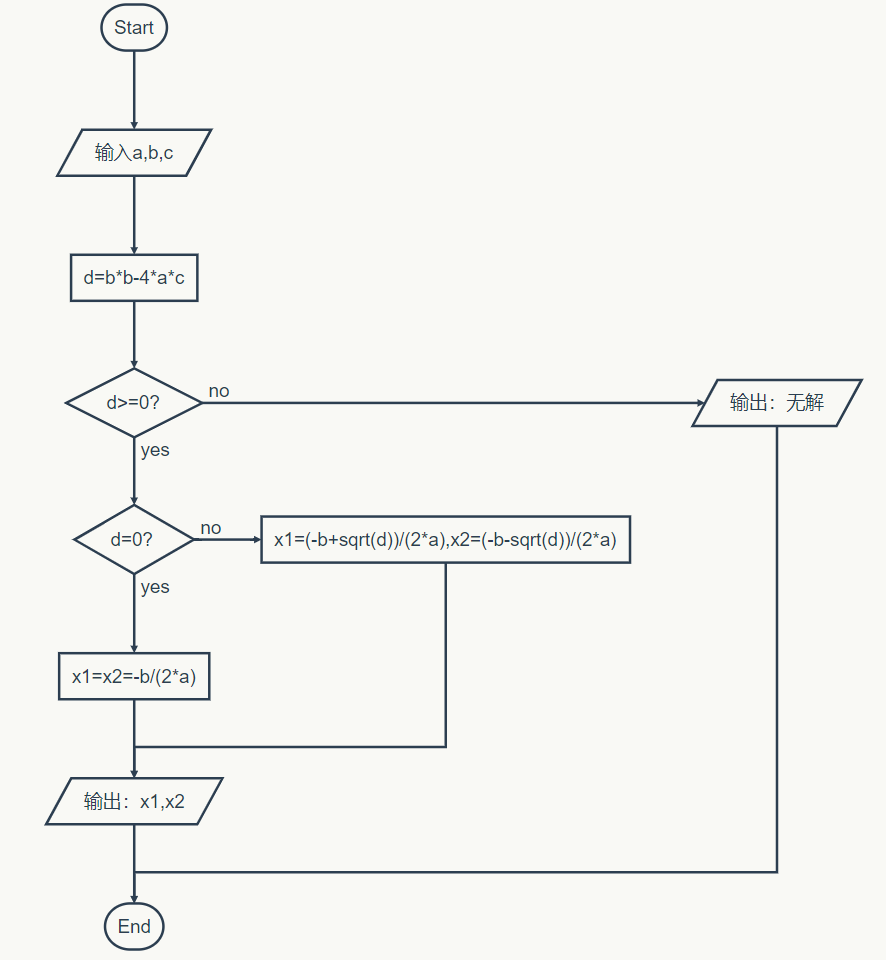
（6）



（7）



（8）



6.

（1）

c=a

a=b

b=c

（2）

n=1

input max

while n< 10 do

input a

if a>max then max=a

n=n+1

end do

print max

（3）

input a,b,c

if a<b then swap(a,b) //交换a,b，具体过程见（1）

if a<c then print c,a,b

else

if c>b then print a,c,b

else print a,b,c

end if

end if

（4）

sum=0

n=1

while n<=100 do

sum=sum+n

n=n+1

end do

print sum

（5）

input n

flag=0

if n mod 3 not = 0 then flag=-1

if n mod 5 not = 0 then flag=-1

if flag=0 then print n “能被3和5整除”

else print n “不能同时被3和5整除”

end if

（6）

n=100

while n<=200 do

i=2

while i<=sqrt(n)

if n mod i =0 then i=n

else i=i+1

end if

end do

if i<sqrt(n) then print n

n=n+1

end do

（7）

input m,n

if m<n then swap(m,n)

r=m mod n

while r not = 0 do

m=n

n=r

r=m mod n

end do

print n

（8）

input a,b,c

delta=b\*b-4\*a\*c

if delta>=0 then

if delta==0 then x1,x2=-b/(2\*a)

else x1=(-b+sqrt(delta))/(2\*a),x2=(-b-sqrt(delta))/(2\*a)

end if

print x1,x2

else print “no solution”

end if