练习6.2.1

假定图**6-26** 中的函数**widen** 可以处理图**6-25a** 的层次结构中的所有类型，翻译下列表达式。假定**d** 是**char** 型，**s** 和**t** 是**short** 型，**x** 是**float** 型。

1. $x = s + d$

```
t1 = (int)s
t2 = (int)d
t3 = t1 + t2
x = (float)t3
```

2. $x = (s + d)/(t + x)$

```
t1 = (int)s
t2 = (int)d
t3 = t1 + t2
t4 = (int)t
t5 = (int)x
t6 = t4 + t5
t7 = t3 / t6
x = (float)t7
```

练习6.2.2

在图**6-36** 的语法制导定义中添加处理下列控制流构造的规则

1. $S->repeat S_1\ until\ B$,当B为真时结束循环

```
S1.next = newlabel()
B.true = S.next
B.false = newlabel()
S.code = label(B.false) || S1.code || label(S1.next) || B.code
```

2. $S->for(S_1;B;S_2)S_3$

```
S1.next = newlabel()
B.true = newlabel()
B.false = S.next
S2.next = S1.next
S3.next = newlabel()
S.code = S1.code
    || lable(S1.next) || B.code
    || lable(B.true) || S3.code
    || label(S3.next) || S2.code
    || gen('goto', S1.next)
```

练习7.1.1

考虑C 语言的函数f 和g：按照图7-7的约定，不考虑编译器优化，讨论当f 调用g 而g 即将返回时运行时栈的状态，其中f 的参数a = 3。只需要讨论返回值、参数、控制链和代码中体现的局部数据。指出

```
int g(int *);
int f(int a){
    int i = a + 2;
    return g(&i);
}
int g(int *b){
    int j= *b;
    return j+2;
}
```

1.哪个函数在栈中为各个元素创建了所使用的空间

函数 f 在栈上为参数 a 和局部变量 i 创建了空间。 函数 g 在栈上为指针参数 b 和局部变量 j 创建了空间。

2.哪个函数写入了各个元素的值？参数、返回值和局部变量的值是什么

函数 f 写入了 a 和 i 的值。a 是函数 f 被调用时传入的参数，值为 3。i 则为 a + 2,结果为 5。 函数 g 写入了 *b和 j 的值。 *b 是从 f 传递的 i 的值，即 。所以 j 是5。 返回值此时是由 g 写入的，j + 2，即 7。

3.这些元素属于哪个活动记录

参数 a、局部变量 i 以及 f 函数调用 g 的返回地址属于 f 的活动记录。 参数 b、局部变量 j 以及 g 返回到 f 的地址属于 g 的活动记录。

练习7.1.2

考虑下面的**Fibonacci**函数：嵌套在**fib0**中的是**fib1**，它假设**n >= 2**并计算第**n**个**Fibonacci**数。嵌套在**fib1**中的是**fib2**，它假设**n >= 4**。请注意，**fib1**和**fib2**都不需要检查基本情况。我们考虑从对**main**的调用开始，直到（对**fib0(1)**的）第一次调用即将返回的时段。

```
fun main(){
    let
        fun fib0(n) =
            let
                fun fib1(n)=
                    let
                        fun fib2(n) =fib1(n-1) + fib1(n-2)
                    in
                        if n >= 4 then fib2(n)
                        else fib0(n-1) + fib0(n-2)
            in
                if n>= 2 then fib1(n)
            end
    in
        fib0(4)
    end;
}
```

1.请描述出当时的活动记录栈，并给出栈中的各个活动记录的访问链。

Real Parameters: None
Main
Access Link: Main
Real Parameters:
Integer n=4
fib0
Access Link: fib0(4) --> main()
Real Parameters:

Integer n=4
fib1
Access Link: fib1(4) --> fib0(4)
Real Parameters:
Integer n=4
fib2
Access Link: fib2(4) --> fib1(4)
Real Parameters:
Integer n=3
fib1
Access Link: fib1(3) --> fib0(4)
Real Parameters:
Integer n=2
fib0
Access Link: fib0(2) --> main()
Real Parameters:
Integer n=2
fib1
Access Link: fib1(2) --> fib0(2)
Real Parameters:
Integer n=1
fib0
Access Link: fib0(1) --> main()

2.假设我们使用**display**表来实现下图中的函数。请给出**fib0(1)**的第一次调用即将返回时的**display**表。同时指明那时在栈中的各个活动记录中保存的**display**表条目。

d[1]-->1	**1. Main**
d[2]-->15	2. fib0(4)

d[1]-->1	1. Main
d[3]-->13	3. d[2]
d[4]-->7	4. fib1(4)
	5. d[3]
	6. fib2(4)
	7. d[4]
	8. fib1(3)
	9. d[3]
	10. fib0(2)
	11. d[2]
	12. fib1(2)
	13. d[3]
	14. fib0(1)
	15. d[2]