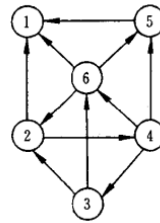


7.1

◆7.1① 已知如右图所示的有向图，请给出该图的



- (1) 每个顶点的入/出度；
- (2) 邻接矩阵；
- (3) 邻接表；
- (4) 逆邻接表；
- (5) 强连通分量。

(1) $OD(1)=0$ $ID(1)=3$ $OD(2)=2$ $ID(2)=2$

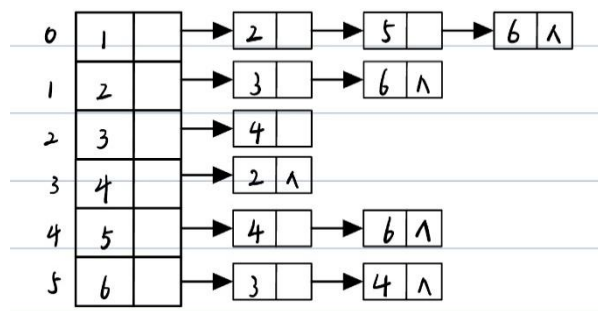
$OD(3)=2$ $ID(3)=1$ $OD(4)=3$ $ID(4)=1$

$OD(5)=1$ $ID(5)=2$ $OD(6)=3$ $ID(6)=2$

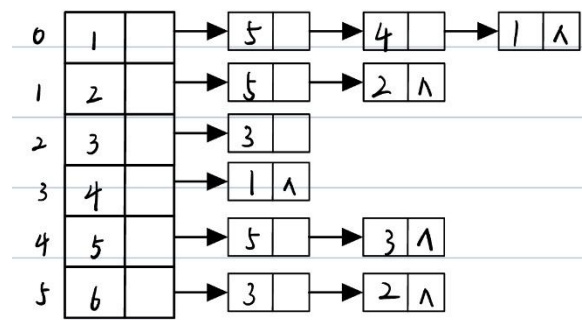
(2) 终点

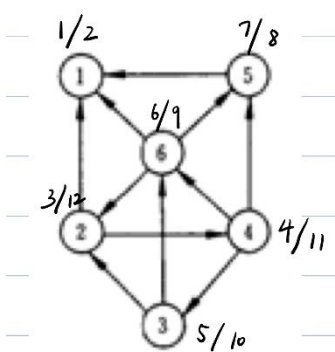
起点 \ 终点	1	2	3	4	5	6
1	0	0	0	0	0	0
2	1	0	0	1	0	0
3	0	1	0	0	0	1
4	0	0	1	0	1	1
5	1	0	0	0	0	0
6	1	1	0	0	1	0

(3) ~~正邻接表~~



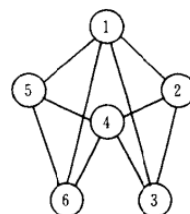
(4) ~~逆邻接表~~



(5)  ①DFS 遍历 G, 如图, 按访问顺序生成逆后序: **243651**
 (标注数对为: 发现顺序/访问顺序)
 ②按逆后序出发对逆向图 G' 进行 DFS 得到以下强
 连通分量: **1、5、2346**

7.3

◆7.3② 画出左图所示的无向图的邻接多重表,使得其中每个无向边结点中第一个顶点号小于第二个顶点号,且每个顶点的各邻接边的链接顺序,为它所邻接到的顶点序号由小到大的顺序。列出深度优先和广度优先搜索遍历该图所得顶点序列和边的序列。



邻接表:

0	1	→ 5	→ 4	→ 2	→ 1	Λ
1	2	→ 3	→ 2	→ 0	Λ	
2	3	→ 3	→ 1	→ 0	Λ	
3	4	→ 5	→ 4	→ 2	→ 1	Λ
4	5	→ 5	→ 3	→ 0	Λ	
5	6	→ 4	→ 3	→ 0	Λ	

邻接多重表:

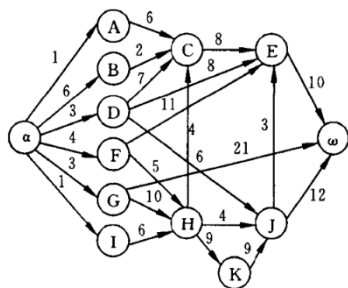
0	1	→ 1	→ 2	→ 2	→ 3	→ 3	→ 4	→ 4	→ 5	→ 5	→ 6	→ 6	→ 0
1	2	→ 1	→ 2	→ 2	→ 3	→ 3	→ 4	→ 4	→ 5	→ 5	→ 6	→ 6	→ 1
2	3	→ 1	→ 2	→ 2	→ 3	→ 3	→ 4	→ 4	→ 5	→ 5	→ 6	→ 6	→ 2
3	4	→ 1	→ 2	→ 2	→ 3	→ 3	→ 4	→ 4	→ 5	→ 5	→ 6	→ 6	→ 3
4	5	→ 1	→ 2	→ 2	→ 3	→ 3	→ 4	→ 4	→ 5	→ 5	→ 6	→ 6	→ 4
5	6	→ 1	→ 2	→ 2	→ 3	→ 3	→ 4	→ 4	→ 5	→ 5	→ 6	→ 6	→ 5

深度优先遍历 顶点序列: 156432 边序列: 15、56、64、43、32

广度优先遍历 顶点序列: 156324 边序列: 15、16、13、12、54

7.10

◆7.10② 对于题 7.10 图所示的 AOE 网络,计算各活动弧的 $e(a_i)$ 和 $l(a_j)$ 函数值、各事件(顶点)的 $ve(v_i)$ 和 $vl(v_j)$ 函数值;列出各条关键路径。



对顶点进行拓扑排序得到顶点序列：

$\alpha > A > B > D > F > G > I > H > C > K > J > E > \omega$

顶点 v	α	A	B	D	F	G	I	H	C	K	J	E	ω
$ve(v)$	0	1	6	3	4	3	1	13	17	22	31	25	44
$vl(v)$	0	20	24	19	8	3	7	13	26	22	31	34	44

v	A	B	D	F	G	I
$e(<\alpha, v>)$	0	0	0	0	0	0
$l(<\alpha, v>)$	19	18	16	4	0	6

$a = \langle A, C \rangle: e(a)=1, l(a)=26-6=20$ $a = \langle B, C \rangle: e(a)=6, l(a)=26-2=24$

$a = \langle D, C \rangle: e(a)=3, l(a)=26-7=19$ $a = \langle D, E \rangle: e(a)=3, l(a)=34-8=26$

$a = \langle D, J \rangle: e(a)=3, l(a)=31-6=25$

$a = \langle F, E \rangle: e(a)=4, l(a)=34-11=23$ $a = \langle F, H \rangle: e(a)=4, l(a)=34-5=29$

$a = \langle G, \omega \rangle: e(a)=43, l(a)=44-21=23$ $a = \langle G, H \rangle: e(a)=4, l(a)=34-5=29$

$a = \langle I, H \rangle: e(a)=1, l(a)=13-6=7$

$a = \langle H, J \rangle: e(a)=13, l(a)=31-4=27$ $a = \langle H, K \rangle: e(a)=13, l(a)=31-9=22$

$a = \langle C, E \rangle: e(a)=17, l(a)=34-8=26$

$a = \langle K, J \rangle: e(a)=22, l(a)=31-9=22$

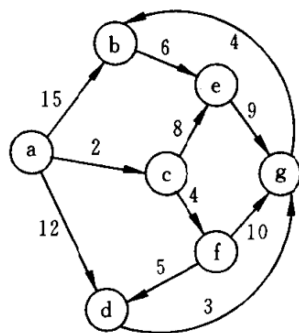
$a = \langle J, \omega \rangle: e(a)=43, l(a)=44-12=32$

$a = \langle E, \omega \rangle: e(a)=43, l(a)=44-10=34$

因此关键路径为： α -G-H-K-J- ω

7.11

◆7. 11② 试利用 Dijkstra 算法求题 7. 11 图中从顶点 a 到其他各顶点间的最短路径，写出执行算法过程中各步的状态。



初始状态如下：

终点	b	c	d	e	f	g
final	0	0	0	0	0	0
dist	15	2	12	∞	∞	∞

终点	b	c	d	e	f	g
final	0	1	0	0	0	0
dist	15	2	12	10	6	∞

终点	b	c	d	e	f	g
final	0	1	0	0	1	0
dist	15	2	11	10	6	16

⇒

终点	b	c	d	e	f	g
final	0	1	0	1	1	0
dist	15	2	11	10	6	16

终点	b	c	d	e	f	g
final	0	1	1	1	1	0
dist	15	2	11	10	6	14

⇒

终点	b	c	d	e	f	g
final	0	1	1	1	1	1
dist	15	2	11	10	6	14

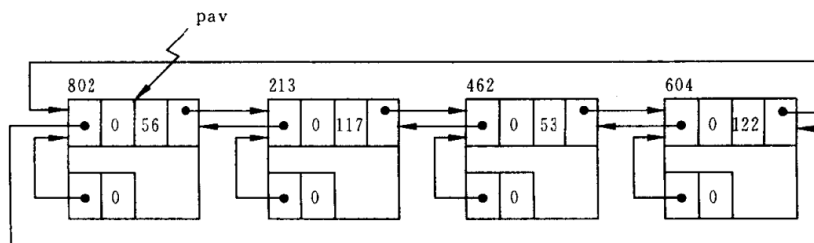
终点	b	c	d	e	f	g
final	1	1	1	1	1	1
dist	15	2	11	10	6	14

故 a 到其他结点的最短路径如下

终点	路径
b	a-b
c	a-c
d	a-c-f-d
e	a-c-e
f	a-c-f
g	a-c-f-g

8.1

◆8.1② 假设利用边界标识法首次适配策略分配,已知在某个时刻的可利用空间表的状态如下图所示:

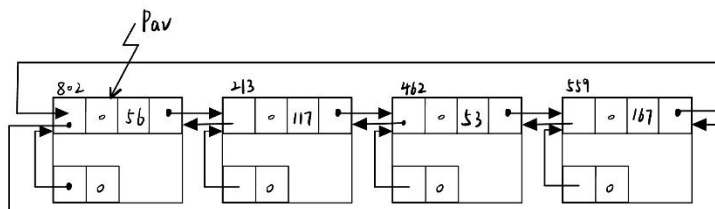


(1) 画出当系统回收一个起始地址为 559、大小为 45 的空闲块之后的链表状态;

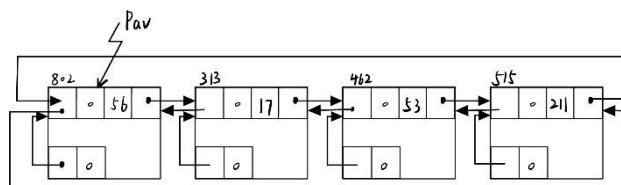
(2) 画出系统继而在接受存储块大小为 100 的请求之后,又回收一块起始地址为 515、大小为 44 的空闲块之后的链表状态。

注意: 存储块头部中大小域的值和申请分配的存储量均包括头和尾的存储空间。

(1)



(2)



8.7

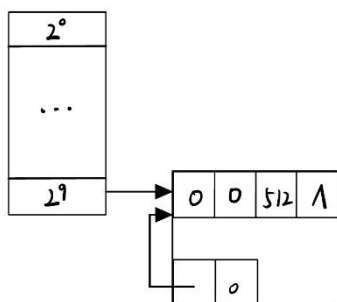
◆8.7③ 已知一个大小为 512 字的内存,假设先后有 6 个用户提出大小分别为 23, 45, 52, 100, 11 和 19 的分配请求,此后大小为 45, 52 和 11 的占用块顺序被释放。假设以伙伴系统实现动态存储管理,

(1) 画出可利用空间表的初始状态;

(2) 画出 6 个用户进入之后的链表状态以及每个用户所得存储块的起始地址;

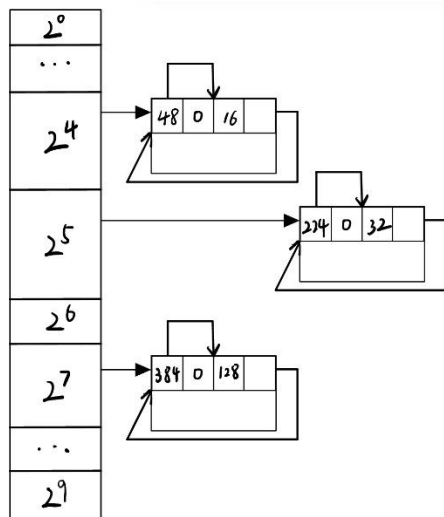
(3) 画出在回收三个用户释放的存储块之后的链表状态。

(1)

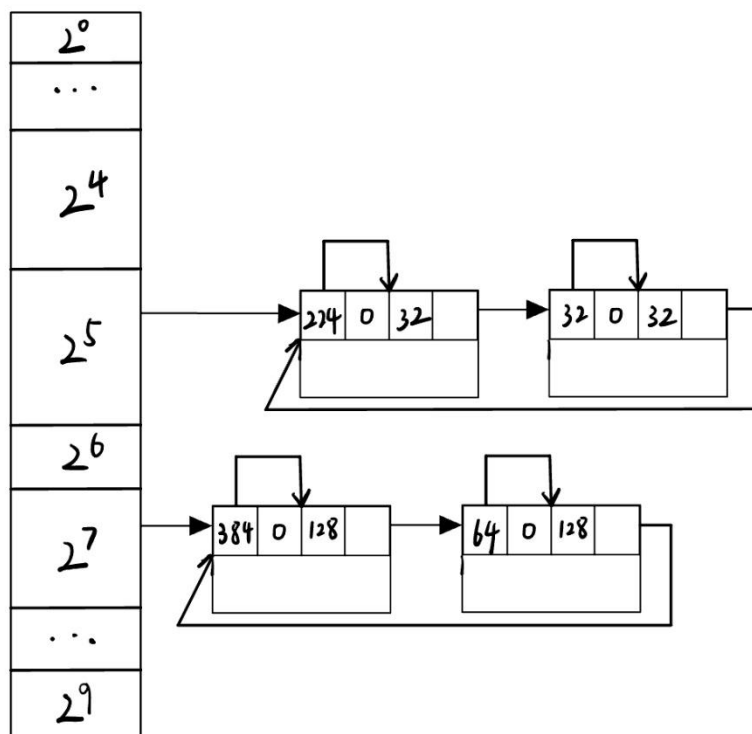


(2)

用户序号	1	2	3	4	5	6
起始地址	0	64	128	256	32	192



(3)



9.1

◆9.1② 若对大小均为 n 的有序的顺序表和无序的顺序表分别进行顺序查找,试在下列三种情况下分别讨论两者在等概率时的平均查找长度是否相同?

- (1) 查找不成功,即表中没有关键字等于给定值 K 的记录;
- (2) 查找成功,且表中只有一个关键字等于给定值 K 的记录;

(3) 查找成功,且表中有若干个关键字等于给定值 K 的记录,一次查找要求找出所有记录。此时的平均查找长度应考虑找到所有记录时所用的比较次数。

(1) 不同。假设有有序表按升序排列,则搜索过程中遇到比 K 大的值则说明没有 K 的记录,可以停止搜索;而无序表需要搜完 n 个元素后才能确定 K 的记录不存在;所以在查找不成功的情况下,有序表的平均查找长度小于无序表的平均查找长度。

(2)


	有序表	无序表	是否相同
ASL	$(1+2+\cdots+n)/n$ $= (n+1)/2$	$(1+2+\cdots+n)/n$ $= (n+1)/2$	相同


(3) 不同, K 在表中有多个记录时且查找成功的情况下,有序表只需要搜索到比 K 大的值时即可以确定找出了所有等于定值 K 的记录,平均查找长度 $< n$ 。而无序表只有搜索完整个表后才能确定找出了所有等于定值 K 的记录,平均查找长度 $= n$ 。故这种情况下有序表的平均查找长度小于无序表的平均查找长度。


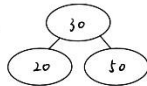
9.14

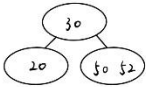
◆9.14② 试从空树开始,画出按以下次序向 2-3 树即 3 阶 B-树中插入关键码的建树过程:20,30,50,52,60,68,70。如果此后删除 50 和 68,画出每一步执行后 2-3 树的状态。

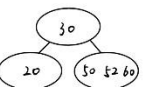
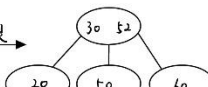
空树 

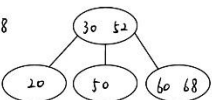
插入 20 

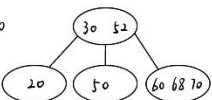
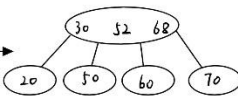
插入 30 

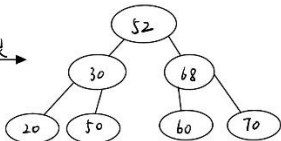
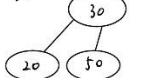
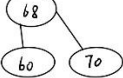
插入 50  分裂 

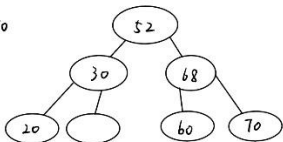
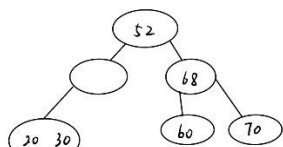
插入 52 

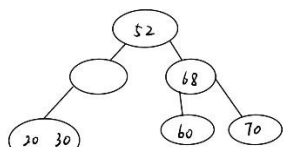
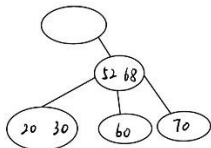
插入 60  分裂 

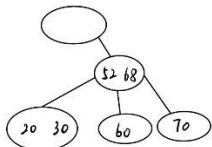
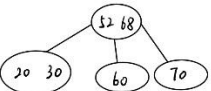
插入 68 

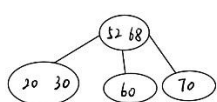
插入 70  分裂 

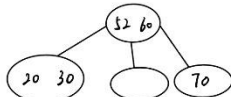
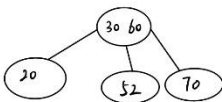
分裂   

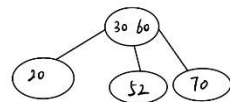
删除 50  合并 

 合并 

 直接删除空根 



删除 68  (前驱替代) 借字 



9.19

◆9.19③ 选取哈希函数 $H(k) = (3k) \text{ MOD } 11$ 。用开放定址法处理冲突, $d_i = i((7k) \text{ MOD } 10 + 1)$ ($i=1, 2, 3, \dots$)。试在 $0 \sim 10$ 的散列地址空间中对关键字序列(22, 41, 53, 46, 30, 13, 01, 67)造哈希表, 并求等概率情况下查找成功时的平均查找长度。

$$H(22) = 66 \text{ MOD } 11 = 0$$

$$H(41) = 123 \text{ MOD } 11 = 2$$

$$H(53) = 159 \text{ MOD } 11 = 5$$

$$H(46) = 138 \text{ MOD } 11 = 6$$

$$H(30) = 90 \text{ MOD } 11 = 2 \text{ (冲突)}$$

$$d_1 = 1 * (210 \text{ MOD } 10 + 1) = 1 \quad H_1(30) = 2 + d_1 = 3$$

$$H(13) = 39 \text{ MOD } 11 = 6 \text{ (冲突)}$$

$$d_1 = 1 * (91 \text{ MOD } 10 + 1) = 2 \quad H_2(13) = 6 + d_1 = 8$$

$$H(01) = 03 \text{ MOD } 11 = 3 \text{ (冲突)}$$

$$d_1 = 1 * (07 \text{ MOD } 10 + 1) = 8 \quad H_1(01) = (3 + d_1) \text{ MOD } 11 = 3 \text{ (冲突)}$$

$$d_2 = 2 * (07 \text{ MOD } 10 + 1) = 16 \quad H_2(01) = (3 + d_2) \text{ MOD } 11 = 0 \text{ (冲突)}$$

$$d_3 = 3 * (07 \text{ MOD } 10 + 1) = 24 \quad H_3(01) = (3 + d_3) \text{ MOD } 11 = 8 \text{ (冲突)}$$

$$d_4 = 4 * (07 \text{ MOD } 10 + 1) = 32 \quad H_4(01) = (3 + d_4) \text{ MOD } 11 = 5 \text{ (冲突)}$$

$$d_5 = 5 * (07 \text{ MOD } 10 + 1) = 40 \quad H_5(01) = (3 + d_5) \text{ MOD } 11 = 2 \text{ (冲突)}$$

$$d_6 = 6 * (07 \text{ MOD } 10 + 1) = 48 \quad H_6(01) = (3 + d_6) \text{ MOD } 11 = 10$$

$$H(67) = 201 \text{ MOD } 11 = 3 \text{ (冲突)}$$

$$d_1 = 1 * (67 * 7 \text{ MOD } 10 + 1) = 10 \quad H_1(67) = (3 + d_1) \text{ MOD } 11 = 2 \text{ (冲突)}$$

$$d_2 = 2 * (67 * 7 \text{ MOD } 10 + 1) = 20 \quad H_2(67) = (3 + d_2) \text{ MOD } 11 = 1$$

综上, 最终哈希表为:

地址	0	1	2	3	4	5	6	7	8	9	10
关键字	22	67	41	30		53	46		13		01
查找次数	1	3	1	2		1	1		2		6

$$(\text{按记录的查找次数计算}) \text{ ASL} = (1 + 1 + 1 + 1 + 2 + 2 + 6 + 3) / 8 = 17 / 8 = 2.125$$

$$(\text{按公式计算}) \text{ 装填因子 } \alpha = 8 / 11, \text{ 线性探测法成功时 } \text{ ASL} \approx 0.5 * (1 + 1 / (1 - \alpha)) = 7 / 3 = 2.333$$

9.24

◆9.24⑤ 某校学生学号由8位十进制数字组成： $c_1c_2c_3c_4c_5c_6c_7c_8$ 。 c_1c_2 为入学时年份的后两位； c_3c_4 为系别：00~24分别代表该校的25个系； c_5 为0或1，0表示本科生，1表示研究生； $c_6c_7c_8$ 为对某级某系某类学生的顺序编号：对于本科生，它不超过199，对于研究生，它不超过049，共有4个年级，四年级学生1996年入学。

(1) 当在校人数达极限情况时，将他们的学号散列到0~24999的地址空间，问装载因子是多少？

(2) 求一个无冲突的哈希函数 H_1 ，它将在校生学号散列到0~24999的地址空间。其簇聚性如何？

(3) 设在校生总数为15000人，散列地址空间为0~19999，你是否能找到一个(2)中要求的 H_1 ？若不能，试设计一个哈希函数 H_2 及其解决冲突的方法，使得多数学号可只经一次散列得到(可设各系各年级本科生平均人数为130，研究生平均人数为20)。

(4) 用算法描述语言表达 H_2 ，并写出相应的查找函数。

$$(1) \text{ 极限人数} = 4 \times 25 \times (199 + 1 + 049 + 1) = 25000$$

$$\text{装载因子} \alpha = 25000 / 25000 = 1$$

$$(2) H_1(c_1c_2c_3c_4c_5c_6c_7c_8) = 1000 \times c_3c_4 + 250 \times c_1c_2 + 200 \times c_5 + c_6c_7c_8$$

簇聚性分析：同一系、同年级、同类别的学生地址连续，簇聚性较高。

(3) 不能找到无冲突的哈希函数。

按系、年级分配地址块，每块有 $20000 / (25 \times 4) = 200$ 个地址，

本科生用前 $200 \times 130 / 150 \approx 174$ (进一法)，研究生用 175-199

$$H_2(c_1c_2c_3c_4c_5c_6c_7c_8) = (c_1c_2 \times 5000 + c_3c_4 \times 200 + c_5 \times 175 + c_6c_7c_8) \bmod 20,000$$

该哈希函数按照各系各年级的本科生与研究生人数之比为双方都预留了空间，尽可能地避免

冲突，多数学号可以一次散列成功。

实际过程中若遇到冲突，可使用线性探测法： $H_i(\text{key}) = (i \times 200 + H_2(\text{key})) \bmod 20,000$ ，

$i = 1, 2, 3, \dots, 24$ 进行跨地址块散列。

(4) 哈希函数描述：

ADDRESS $H_2(\text{key})\{$

```

c1c2=key/1000000, c3c4=(key/10000)%100;

c5=(key/1000)%10, c6c7c8=key%1000;

return (c1c2×5000+c3c4×200+c5×175+c6c7c8) %20,000

}

```

查找算法描述：（设哈希表无记录的位置为 NONE）

```

ADDRESS HashSearch(key, HList){

    address = H2(key);

    int i = 0; // 已进行的探测次数

    while(HList[address] != key){

        if(HList[address] == NONE || i>=24) return NONE;// 最多探测 24 次

        i++;

        address = (address + 200) % 20000;

    }

    return address;

}

```