一、问题分析

- 分析并确定要处理的对象（数据）是什么
  从键盘输入一个整数和一组整数，那个整数等于后一组元素的个数
- 分析并确定要实现的功能是什么
  1、从键盘接受一个数，与一组整数
  2、将一组整数从小到大排序
  3、比较相邻两元素的之间差值的绝对值
  4、输出相邻两元素间的差值的绝对值的最小值
- 分析并确定处理后的结果如何显示
  直接在屏幕上输出两元素间差值的绝对值的最小值
- 请用题目中样例，详细给出样例求解过程。
  输入数据：5　　1 5 4 8 20
  （1,5,4,8,20）=>A
  min(abs(A[i]-A[j]))=>1

二、数据结构和算法设计

- **抽象数据类型设计**

数据对象：一组整数
数据关系：一组整数从键盘依次输入，按照输入的先后次序，满足线性关系
基本操作：准备能够存储一组整数的存储空间
插入元素
交换元素位置
遍历空间，得到存储的值
对一组整数排序
返回线性表长度

- **物理数据对象设计**

抽象数据类型：线性表
物理数据类型：集合元素是整数，物理数据类型用整形变量 int 的先线性表，因多次涉及对元素位置的交换，所以选用基于链表实现的线性表。

ADT IntegerSet{
数据对象：　D = { ai |ai∈整数，i= 1,2…, n, n>=0}
数据关系：　R = {<ai-1，ai> |ai-1, ai∈　D}
基本操作：
void init( )
bool push_back (const Elem&)
void get_head( )
void next( )
void getval(Elem&)
int getLength ( )
void sort()
void swap()

}

- **算法思想的设计**

本程序由三个模块组成：

- **输入模块：**设计一个把输入的整数集合，保存为线性表的函数 void input（List<int>&L）,完成一组整数的输入；
- **计算模块：**设计一个求这组整数两元素间最小差值的模块 int count（List<int>&L）；
- **输出模块:**输出计算模块中返回的值；

- **关键功能的算法步骤**

输入模块：

1、从键盘读入数据量 n；
2、从键盘读取整数
3、将整数添加入线性表

```
void input(Linklist<int>& list)
{
	int n;
	cin >> n;//接受数据量
	for (int i = 0; i < n; i++)
	{
		int a;
		cin >> a;
		list.push_back(a);//添加元素
	}
	return;
}
```

计算模块：

1、对线性表元素排序
2、遍历线性表，寻找两元素间差值的最小值
3、返回最小值

```
int count(Linklist<int>& list)
{
	list.sort();//将链表元素由小到大排序
	Link<int>* p = list.get_head();
	int min = abs(p->getval() - p->getnext()->getval());
	for (p = list.get_head(); p != list.get_tail()->getpre(); p = p->getnext())
	{
		if (abs(p->getval() - p->getnext()->getval()) < min)//寻找比之前出现的两元素只差绝对值更小的元素差的绝对值
			min = abs(p->getval() - p->getnext()->getval());
	}
	return min;//返回元素间差值绝对值的最小值
```

}

输出模块：cout<<min;

三、算法性能分析

输入模块：函数 void input(Linklist<int>& list)时间复杂度为 O(n);
计算模块：函数 int count(Linklist<int>& list) 时间复杂度为 O(nlog(n))
输出模块：函数 cout，时间复杂度为 O(1)