# HW5 林宸昊 PB20000034

## 5.1

- (1) 体积为6x8x6=288个字节;
- (2) 地址为 1000 + (6 x 8 x 6 - 6) = 1282;
- (3) 地址为1000 + (8 + 4) x 6 = 1072;
- (4) 地址为1000 + (6 x 7 + 4) x 6 = 1276;

## 5.2

- i is odd: k = i + j - 2;
- i is even: k = i + j - 1;
- that is k = i + j - (i % 2) - 1;

## 5.17

```
void max(List a, int n, int &s){
    if(n == a.length)
        return ;
    else{
        if(s < a.elem[n]) s = a.elem[n];
        max(a, n + 1, s);
    }
}

void min(List a, int n, int &s){
    if(n == a.length)
        return ;
    else{
        if(s > a.elem[n]) s = a.elem[n];
        min(a, n + 1, s);
    }
}

void sum(List a, int n, int &s){
    if(n == 0) s = 0;
    if(n == a.length)
        return ;
    else{
        s += a.elem[n];
        sum(a, n + 1, s);
    }
}

void multi(List a, int n, int &s){
    if(n == 0) s = 1;
    if(n == a.length)
        return ;
    else{
        s *= a.elem[n];
```

```
        multi(a, n + 1, s);
    }
}
//该算法会丢失大量精度，最好递归求总和再求平均
void aver(List a, int n, float &s){
    if(n == 0) {
        s = a.elem[0];
        aver(a, n + 1, s);
    }
    if(n == a.length)
        return ;
    else {
        s = n * s + a.elem[n];
        s /= (n + 1);
        aver(a, n + 1, s);
    }
}
```

5.19

```
void SaddlePoint(Elemtype matrix[][n], int col, int row){
    //先储存每行每列的最大最小值
    Elemtype row_min[row], col_max[col];
    //初始化数组
    for(int i = 0;i < row;i++){
        row_min[i] = matrix[i][0];
    }
    for(int i = 0;i < col;i++>){
        col_max[i] = matrix[0][i];
    }
    for(int i = 0;i < row;i++){
        for(int j = 0;j < col;j++){
            if (row_min[i] > matrix[i][j]) row_min[i] = matrix[i][j];
            if (col_max[j] < matrix[i][j]) col_max[j] = matrix[i][j];
        }
    }
    //寻找马鞍点
    for(int i = 0;i < row;i++){
        for(int j = 0;j < col;j++){
            if(row_min[i] == col_max[j]){
                printf("matrix[%d][%d] = %() is saddlepoin\n", i, j,
matrix[i][j]);
            }
        }
    }
}
```

- 最坏情况下时间复杂度为O(row*col).