**Practice 1: Locality Optimizing**

**SOLUTION**

void smooth(int dim, pixel \*src, pixel \*dst) {

int i, j;

for(i=0; i<dim;i++) {

COPY(&dst[PIXEL(i,0,dim)], &src[PIXEL(i,0,dim)]);

COPY(&dst[PIXEL(i,dim-1,dim)], &src[PIXEL(i,dim-1,dim)]);

}

for(j=1; j<dim-1;j++) {

COPY(&dst[PIXEL(0,j,dim)], &src[PIXEL(0,j,dim)]);

}

for (j = 1; j < dim - 1; j++) {

COPY(&dst[PIXEL(dim - 1, j, dim)], &src[PIXEL(dim - 1, j, dim)]);

}

int div = 3;

for(j=1; j<dim-1; j++) {

for(i=1; i<dim-1; i++) {

SMOOTH(&dst[PIXEL(j,i,dim)],

&src[PIXEL(j,i,dim)],

&src[PIXEL(j-1,i,dim)],

&src[PIXEL(j+1,i,dim)],

&src[PIXEL(j,i+1,dim)],

&src[PIXEL(j,i-1,dim)],

&src[PIXEL(j-1,i-1,dim)],

&src[PIXEL(j+1,i+1,dim)],

&src[PIXEL(j-1,i+1,dim)],

&src[PIXEL(j+1,i-1,dim)]);

}

}

return;

}

void rotate(int dim, pixel \*src, pixel \*dst) {

int i, j,k,l;

int div = 4;

for(i=0; i < dim; i+=div) {

for(j=0; j < dim; j++) {

for (k=i;k<i+div;k++)

//for (l=j;l<j+div;l++)

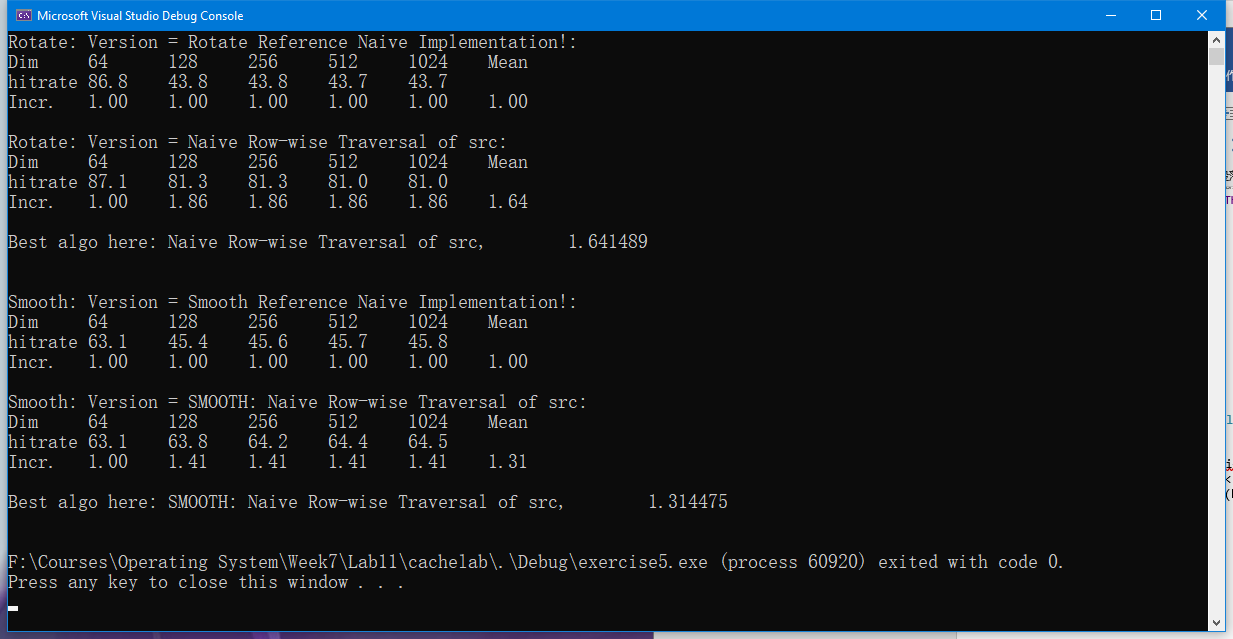
COPY(&dst[PIXEL(dim-1-j,k,dim)], &src[PIXEL(k,j,dim)]);

}

}

return;

}



**Practice 2: Draw Memory Mountain**

1. What is the variable of size and stride used for?

让我们控制产生出的读序列的时间和空间局部性程序.

1. What is the meaning of the return value of the function run()?

内存吞吐量程度

1. How many ridges(山脊) do you get? And what are they correspond to?

3 , L1cache , L2 cache, L3 cache

1. Just compare the mountain with the information that you just got by running as everest.exe. And explain the How the spatial locality and temporal locality are shown in this figure.

Size的值越小,得到的工作集越小,因此时间局部性越好,stride值越小,得到的空间局部性越好.