#include <stdio.h>

#include <stdlib.h>

int main(int argc, char \* argv[]) {

int a1 = 10;

int a2 = 10;

double b1 = 3.14;

double b2 = 9.8;

short c1 = 100;

short c2 = 20;

char d1 = 'a';

char d2 = 'c';

int M = 4;

int N = 6;

int A[M][N];

int i, j;

//You can find the memory address of a function by using & operator

printf("------------------- The address of main function: %p ---------------------------\n", &main);

printf("--- Variable memory addresses using & operator and variable sizes using sizeof operator ---\n");

/\*\* TODO #1 (10 points): add your code for printing addresses and sizes for variables a1, a2, b1, b2, c1, c2, d1, d2 \*/

printf("Adress of a1: %p and size %zu\n", &a1, sizeof(a1));

printf("Adress of a2: %p and size %zu\n", &a2, sizeof(a2));

printf("Adress of b1: %p and size %zu\n", &b1, sizeof(b1));

printf("Adress of b2: %p and size %zu\n", &b2, sizeof(b2));

printf("Adress of c1: %p and size %zu\n", &c1, sizeof(c1));

printf("Adress of c2: %p and size %zu\n", &c2, sizeof(c2));

printf("Adress of d1: %p and size %zu\n", &d1, sizeof(d1));

printf("Adress of d2: %p and size %zu\n", &d2, sizeof(d2));

printf("\n");

printf("--- Memory addresses of array elements using & operator and base+offset calcualtion ----\n");

/\*\* TODO #2 (20 points): add your code for printing addresses of array elements using & operator and base+offset calcualtion \*/

printf("Base adress of array A[4][6]: %p\n", A);

for(i=0; i<4; i++) {

for(j=0; j<6; j++) {

int offset = sizeof(int)\*(i\*N+j);

printf("Memory Address of &A[%d][%d]: %p, offset: %04x, base+offset: %p\n", i, j, (char\*)A+sizeof(int)\*(i\*4 +j), offset, (char\*)A+offset);

printf("\n");

}

}

/\* 1-D stencil operation: for an array B[M], update each element by B2[i] = (B[i-1]+B[i]+B[i+1])/3 \*/

srand(1<<12); // Initialize random number generator seed, should only be called once.

M = 100;

int B[M];

int \*iterator = B;

//generate rand number for array B and print array B

printf("\n------------------------- 1-D stencil operation --------------------------------------\n");

printf("Element values of array B[%d]\n", M);

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

//TODO #3 (10 points): update the iterator to store the address of element i of B.

iterator = B+i;

\*iterator = rand() % 20; /\* assign the array element a random value between 0 and 20 \*/

printf("%d\t", \*iterator);

if ((i+1)%10==0) printf("\n"); //go to the next line

}

iterator = B;

int B2[M];

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

/\* TODO #4 (35 points): perform operation B2[i] = (B[i-1]+B[i]+B[i+1])/3. You are only allowed to use

\* the iterator and i variable to calcualte the memory addresses of needed elements of B and B2.

\* You should NOT use [] or & operator for any purpose here \*/

iterator = B+i;

\*(B2+i) = (\*(iterator -1) + \*(iterator) + \*(iterator + 1))/3;

}

/\* boundary copy \*/

\*B2 = \*B;

\*(B2+M-1) = \*(B+M-1);

printf("\nElement values of array B2[%d] after 1-D stencil operation on array B\n", M);

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

//TODO #5 (5 points): update the iterator to store the address of element i of B2.

iterator = B2+i;

printf("%d\t", \*iterator);

if ((i+1)%10==0) printf("\n"); //go to the next line

}

return 0;

}



