DMP: Дамп памяти

Код. Шаги 0-3

```c
#include <stdio.h>


void FillInt(int arr[], int size){
    int number = 600;

    for(int i = 0; i < size; i++){
        arr[i] = number + 2 * i;
    }
}


void FillDouble(double arr[], int size){
    double number = 600;

    for(int i = 0; i < size; i++){
        arr[i] = number + 2 * i;
    }
}


void PrintInt(int arr[], int size){
    for(int i = 0; i < size; i++){
        printf("%d ", arr[i]);
    }

    printf("\n");
}



void PrintDouble(double arr[], int size){
    for(int i = 0; i < size; i++){
        printf("%lf ", arr[i]);
    }

    printf("\n");
}


void MemoryDump(void const* ptr, int size){
    int const byte_per_row = 16;
    int row_count = size / byte_per_row;
    if(size % byte_per_row > 0){
        row_count++;
    }

    for(int i = 0; i < row_count; i++){
        int n_rest_bytes = size - i * byte_per_row;
        int n_current_bytes = n_rest_bytes < byte_per_row ? n_rest_bytes :
byte_per_row;

        printf("%p: ", ptr);
        for(int j = 0; j < n_current_bytes; j++){
            printf("%02x ", *(char*)ptr);
```

```
            ptr += 1;
        }

        printf("\n");
    }
}


int main(void){
    int N = 9;
    int A[N];
    double B[N];

    FillInt(A, N);
    FillDouble(B, N);

    printf("Int array:\n");
    PrintInt(A, N);
    MemoryDump(A, sizeof(A));

    printf("\n");

    printf("Double array:\n");
    PrintDouble(B, N);
    MemoryDump(B, sizeof(B));

    return 0;
}
```

Пример работы программы.

```
Int array:
600 602 604 606 608 610 612 614 616
0x16bd56f20: 58 02 00 00 5a 02 00 00 5c 02 00 00 5e 02 00 00
0x16bd56f30: 60 02 00 00 62 02 00 00 64 02 00 00 66 02 00 00
0x16bd56f40: 68 02 00 00

Double array:
600.000000 602.000000 604.000000 606.000000 608.000000 610.000000 612.000000 614.000000 616.000000
0x16bd56ed0: 00 00 00 00 00 ffffffc0 ffffff82 40 00 00 00 00 00 ffffffd0 ffffff82 40
0x16bd56ee0: 00 00 00 00 00 fffffffe0 ffffff82 40 00 00 00 00 00 fffffff0 ffffff82 40
0x16bd56ef0: 00 00 00 00 00 00 ffffff83 40 00 00 00 00 00 10 ffffff83 40
0x16bd56f00: 00 00 00 00 00 20 ffffff83 40 00 00 00 00 00 30 ffffff83 40
0x16bd56f10: 00 00 00 00 00 40 ffffff83 40
```