Given the following code segments, annotate the code to describe what changes can be made to the code to improve cache memory access time (both data and instruction), memory consumption, and branching, as well as why the change results in the improvement. (You do not need to re-write code for full credit).

```
#include <time.h>
#include <stdio.h>
#include <stdlib.h>

typedef struct example{
    int int1;
    double double1;
    float float1;
    double double2;
    int int2;
}example;

int func( int count, example* the_arr, long unsigned int index ){
    return count += the_arr[index].int1;
}

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

    int array_size = atoi( argv[1] ), sum = 0;

    example* the_list = (example *)calloc(array_size,sizeof(example));

    int i;
    for(i = 0; i < array_size; ++i){
        the_list[i].int1 = i;
        the_list[i].double1 = (double)i;
        the_list[i].float1 = (float)i;
        the_list[i].double2 = (double)(i+1);
        the_list[i].int2 = i+1;
    }

    int test;
    for(test = 0; test < 4; ++test){

        long unsigned int index;
        for( index = 0; index < array_size; ++index ){

            the_list[index].int1 += func( test, the_list, index );
            sum += the_list[index].int2;
        }
    }

    free( the_list );

    return EXIT_SUCCESS;
}
```

① Rearrang Struct
Improving Spatial locality
reduces cache misses

② Re-order loops
Improves temporal locality
to reduce cache misses

③ Intermediate register
reduce lw/sw

④ Loop unrolling
reduces branch misprediction

⑤ Modify function
reduce lw/sw from
struct

⑥ C Macro
Insert assembly to
reduce jal, jalr,
and argument registers

Insert the following elements into a Splay Tree

15, 12, 10, 13, 22, 14, 11,

Then, find the following

13, 17, and 13