```c
/*********************************************************
 * structures: learn how to use and typedef structures *
 *********************************************************/

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

// define a new type particle
typedef struct _particle {
    double r[3];
    int charge;
} particle;

void print_particle(particle atom){
    printf("Position: (%.2f %.2f %.2f), charge: %d\n", atom.r[0], atom.r[1], atom.r[2], atom.charge);
}

// utilise the operator -> in order to avoid (*pt).
void set_values_particle(particle* pt, double x, double y, double z, int c){
    pt -> r[0] = x;
    pt -> r[1] = y;
    pt -> r[2] = z;
    pt -> charge = c;
}

// driver program
int main(){
    particle ion = {.r[0] = 3.,
                    .r[1] = 2.,
                    .r[2] = 1.,
                    .charge = 25 };

    print_particle(ion);
    printf("A particle needs %zu Bytes of storage\n", sizeof(particle));

    set_values_particle(&ion, 2., 2., 2., 2);
    print_particle(ion);

    particle* heap_particle = malloc(sizeof(particle));
    set_values_particle(heap_particle, 1., 2., 3., 4);
    print_particle(*heap_particle);

    free(heap_particle);

    return 0;
}
```