particle.hpp

```cpp
 1  #ifndef PARTICLE_HPP
 2  #define PARTICLE_HPP
 3
 4  #include "particle_type.hpp"
 5  #include "resonance_type.hpp"
 6
 7  #include <array>
 8  #include <cmath>
 9
10  class Particle {
11   public:
12    double get_px() const;
13    double get_py() const;
14    double get_pz() const;
15    char   get_name() const;
16    int    get_index() const;
17    void   set_index(char name);
18    double get_mass() const;
19    int    get_charge() const;
20    double get_energy() const;
21    void   set_p(double px, double py, double pz);
22
23    double invMass(const Particle& other) const;
24
25    static void addParticleType(char name, double mass, int charge,
26                                double width = 0);
27    static void printParticleTypes();
28
29    void printParticle() const;
30
31    int decay2body(Particle& dau1, Particle& dau2) const;
32
33    Particle();
34    Particle(char name, double px = 0, double py = 0, double pz = 0);
35
36   private:
37    static const int max_n_particle_type_ = 7;
38    static std::array<ParticleType*, max_n_particle_type_> particle_types_;
39    static int                                             n_particle_type_;
40
41    int index_;
42
43    double px_;
44    double py_;
45    double pz_;
46
47    void boost(double bx, double by, double bz);
48
```

```cpp
49    static int
50        findParticle(char name); // trova il tipo di particella a partire dal suo
51                            // nome (serve a settare correttamente l'indice)
52   // puntatore perché può restituire un valore nullo se il nome non esiste
53 };
54
55 inline double Particle::get_px() const { return px_; }
56 inline double Particle::get_py() const { return py_; }
57 inline double Particle::get_pz() const { return pz_; }
58
59 inline char Particle::get_name() const {
60   return particle_types_[index_]->get_name();
61 }
62
63 inline void Particle::set_p(double px, double py, double pz) {
64   px_ = px;
65   py_ = py;
66   pz_ = pz;
67 }
68
69 inline int  Particle::get_index() const { return index_; }
70 inline void Particle::set_index(char name) {
71   const int find_particle = findParticle(name);
72   if (find_particle >= 0) { index_ = find_particle; }
73 }
74
75 inline double Particle::get_mass() const {
76   return particle_types_[index_]->get_mass();
77 }
78
79 inline int Particle::get_charge() const {
80   return particle_types_[index_]->get_charge();
81 }
82
83 inline double Particle::get_energy() const {
84   double mass = get_mass();
85   double p2   = px_ * px_ + py_ * py_ + pz_ * pz_;
86   return std::sqrt(mass * mass + p2);
87 }
88
89 inline double Particle::invMass(const Particle& other) const {
90   double other_px = other.get_px();
91   double other_py = other.get_py();
92   double other_pz = other.get_pz();
93
94   double px2 = (px_ + other_px) * (px_ + other_px);
95   double py2 = (py_ + other_py) * (py_ + other_py);
96   double pz2 = (pz_ + other_pz) * (pz_ + other_pz);
97
98   double e12 = get_energy() + other.get_energy();
```

```cpp
 99
100      return std::sqrt(e12 * e12 - px2 - py2 - pz2);
101  }
102
103  #endif
```