Chessman.h

```
01: #ifndef CHESSMAN
02: #define CHESSMAN
03:
04: #include <iostream>
05:
06: class Figure
07: {
08:   protected:
09:     char pos[4];
10:   public:
11:     Figure(char*);
12:     Figure() {};
13:     virtual int attack(char*)=0;
14:     virtual char isa()=0;
15:     operator char*();
16:     static int deskout(char*);
17:     int danger();
18:     Figure& dodge(Figure*);
19:     int city(char*);
20:     Figure& operator=(char*);
21:     int operator==(char*);
22:     int operator!=(char*);
23:     friend std::ostream& operator<<(std::ostream&, Figure&);
24:     friend std::istream& operator>>(std::istream&, Figure&);
25: };
26: #endif
```

Chessman.cc

```
01: #include "chessman.h"
02: using std::cout;
03:
04: Figure::Figure(char* p)
05: {
06:   pos[0] = p[0]; pos[1] = p[1]; pos[2] = pos[3] = '\0';
04: }
05:
06: Figure::operator char*()
07: {
08:   return pos;
09: }
10:
11: int Figure::deskout(char* p)
12: {
13:   return((p[0] > 'h') || (p[0] < 'a') || (p[1] < '1') || (p[1] > '8'));
14: }
15:
16: Figure& Figure::operator=(char* p)
17: {
18:   pos[0] = p[0]; pos[1] = p[1];
19:   return(*this);
20: }
21:
22: int Figure::operator==(char* p)
23: {
24:   return((pos[0] == p[0]) && (pos[1] == p[1]));
25: }
26:
27: int Figure::operator!=(char* p)
28: {
```

```
29:     return((pos[0] != p[0]) || (pos[1] != p[1]));
30:   }
31:
32:   std::ostream& operator<<(std::ostream& out, Figure& f)
33:   {
34:     return out << f.isa() << f.pos[0] << f.pos[1];
35:   }
36:
37:   std::istream& operator>>(std::istream& in, Figure& f)
38:   {
39:     char s[2];
40:     std::cin.unsetf(std::ios::skipws);
41:     in >> s[0] >> s[1];
42:     in.ignore(64, '\n');
43:     if((f.attack(s) == 0) || (f == s))    // if(f.deskout(s))
44:       in.clear(std::ios::failbit | in.rdstate());
45:     f = s;
46:     return in;
47:   }
48:
49:   int Figure::danger()
50:   {
51:     int i, j;
52:     char s[2];
53:     char* mark = ".+x";
54:     char m;
55:     cout << "  a b c d e f g h\n";
56:     for(i=8; i > 0; i--)
57:     {
59:       cout << i << ' ';
60:       s[1] = '0' + i;
61:       for(j=0; j < 8; j++)
62:       {
63:         s[0] = 'a' + j;
64:         m = (*this != s) ? mark[attack(s)] : isa(); // attack ?
65:         cout << m << ' ';
66:       }
67:       cout << i << "\n";
68:     }
69:     cout << "  a b c d e f g h\n";
70:     return(0);
71:   }
72:
73:   int Figure::city(char* p)
74:   {
75:     int dx, dy;
76:     if((dx = pos[0] - p[0]) < 0)
77:       dx = -dx;
78:     if((dy = pos[1] - p[1]) < 0)
79:       dy = -dy;
80:     return(dx + dy);
81:   }
82:
83:   Figure& Figure::dodge(Figure* f)
84:   {
85:     char s[2];
86:     char p[2];
87:     char q[2];
88:     short d;
89:     short dm = 16;
```

```
90:     short i, j;
91:     p[0] = p[1] = q[0] = q[1] = 0;
92:     for(i=8; i > 0; i--)
93:     {
94:       s[1] = '0' + i;
95:       for(j=0; j < 8; j++)
96:       {
97:         s[0] = 'a' + j;
98:         if(*this == s)
99:           continue;
100:        if(!attack(s))
101:          continue;
102:        q[0] = s[0]; q[1] = s[1];
103:        if(f->attack(s))
104:          continue;
105:        if((d = f->city(s)) > dm)
106:          continue;
107:        dm = d; p[0] = s[0]; p[1] = s[1];
108:      }
109:    }
110:    *this = p[0] ? p : q;
110:    return(*this);
111: }
```

Chesslib.h

```
01:  #ifndef CHESSLIB
02:  #define CHESSLIB
03:
04:  #include "chessman.h"
05:
06:  class Plus : virtual public Figure
07:  {
08:    public:
09:      Plus(char* p) : Figure(p) {};
10:      Plus() {};
11:      char isa() { return 'P'; };
12:      int attack(char*);
13:  };
14:
15:  class Cros : virtual public Figure
16:  {
17:    public:
18:      Cros(char* p) : Figure(p) {};
19:      Cros() {};
20:      char isa() { return 'C'; };
21:      int attack(char*);
22:  };
23:
24:  class Star : public Plus, public Cros
25:  {
26:    public:
27:      Star(char* p) : Figure(p) {};
28:      Star() {};
29:      char isa() { return 'S'; };
30:      int attack(char*);
31:  };
32:  #endif
```

Chesslib.cc

```
01:  #include "chesslib.h"
02:
03:  int Plus::attack(char* p)
04:  {
05:    if(deskout(p) > 0)
06:      return(0);
07:    int x = p[0] - pos[0];
08:    int y = p[1] - pos[1];
09:    if(x < 0) x = -x;
10:    if(y < 0) y = -y;
11:    if((x + y) < 2)
12:      return(1);
13:    return(0);
14:  }
15:
16:  int Cros::attack(char* p)
17:  {
18:    if(deskout(p) > 0)
19:      return(0);
20:   int x = p[0] - pos[0];
21:   int y = p[1] - pos[1];
22:   if(x < 0) x = -x;
23:   if(y < 0) y = -y;
24:   if(((x + y) > 2) || ( x != y))
25:     return(0);
26:   return(1);
27:  }
28:
29:  int Star::attack(char* s)
30:  {
31:    if(Plus::attack(s) > 0)
32:      return(1);
33:    if(Cros::attack(s) > 0)
34:      return(2);
35:    return(0);
36:  }
```

chess+.cc

```
01:  #include <iostream>
02:  #include "chesslib.h"
03:
04:  int main(int argc, char* argv[])
05:  {
06:    if(argc < 2)
07:      argv[1] = "nil";
08:    if(Figure::deskout(argv[1]))
09:    {
10:      std::cout << "Correct " << argv[1];
11:      std::cout << " to " << (argv[1] = "a1") << " Default\n";
12:      std::cout << "Usage example: chess+ e5\n";
13:    }
14:    Star f(argv[1]);
15:    do
16:    {
17:      f.danger();
18:      std::cout << f << '-' << f.isa();
19:    } while (std::cin >> f);
20:    return(0);
21:  }
```