Esercizio di traduzione del 5 febbraio 2020

Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

```
struct st1 { int vi[4]; };
struct st2 { char vd[4]; };
class cl {
        long v2[4]; char v1[4]; int v3[4];
public:
        cl(st1 ss);
        cl(st1 s1, int ar2[]);
        cl elab1(const char *ar1, st2& s2);
        void stampa() {
                for (int i = 0; i < 4; i++) cout << (int)v1[i] << ' '; cout << endl;
                for (int i = 0; i < 4; i++) cout << (int)v2[i] << ' '; cout << endl;
                for (int i = 0; i < 4; i++) cout << (int)v3[i] << ' '; cout << endl << endl;
        }
};
```

+8	+7	+6	+5	+4	+3	+2	+1	
v2[0]								+0
v2[1]								+8
v2[2]								+16
v2[3]								+24
v3[0]				v1[3]	v1[2]	v1[1]	v1[0]	+32
v3[2]				v3[1]				+40
X	X	X	X	v3[3]				+48

Realizzare in Assembler GCC le funzioni membro seguenti.

```
cl::cl(st1 ss)
{
        for (int i = 0; i < 4; i++) {
                v1[i] = ss.vi[i]; v2[i] = ss.vi[i] * 2;
                v3[i] = 4 * ss.vi[i];
        }
}
cl::cl(st1 s1, int ar2[])
{
        for (int i = 0; i < 4; i++) {
                v1[i] = s1.vi[i]; v2[i] = s1.vi[i] * 8;
                v3[i] = ar2[i];
        }
}
cl cl::elab1(const char *ar1, st2& s2)
{
        st1 s1;
        for (int i = 0; i < 4; i++) s1.vi[i] = ar1[i] + i;
        cl cla(s1);
        for (int i = 0; i < 4; i++) cla.v3[i] = s2.vd[i];
        return cla;
}
```

_ZN2clC1E3st1

_ZN2clC1E3st1Pi

Primo parametro delle funzioni: puntatore implicito *this**

_ZN2clC1EKPcR3st2 -> _ZN2clC1EPcR3st2

Registri scratch: RAX, R10, R11, *RCX, RDX, RSI, RDI, R8, R9*.
Registri non scratch: RBP, RBX, R12, R13, R14, R15.
Registri per i parametri in ingresso: RDI, RSI, RDX, RCX, R8, R9
Registri per il valore di ritorno: RDX_RAX

Esercizio di traduzione del 16 gennaio 2020

Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

```
struct st1 { long vc[4]; }; struct st2 { int vd[4]; };
class cl {
        st1 s;
        char v[4];
public:
        cl(const long *c, st2 s2);
        void elab1(st1& s1, st2 s2);
        void stampa()
        {       int i;
                for (i=0;i<4;i++) cout << s.vc[i] << ' '; cout << endl;
                for (i=0;i<4;i++) cout << (int)v[i] << ' '; cout << endl << endl;
        }
};
```

+8	+7	+6	+5	+4	+3	+2	+1	
s.vc[0]								+0
s.vc[1]								+8
s.vc[2]								+16
s.vc[3]								+24
X	X	X	X	v[3]	v[2]	v[1]	v[0]	+32

Realizzare in Assembler GCC le funzioni membro seguenti.

```
cl::cl(const long *c, st2 s2)
{
        for (int i = 0; i < 4; i++) {
                s.vc[i] = c[i];
                v[i] = s2.vd[i] + s.vc[i];
        }
}
void cl::elab1(st1& s1, st2 s2)
{
        cl cla(s1.vc, s2);
        for (int i = 0; i < 4; i++) {
                if (s.vc[i] < s1.vc[i])
                        s.vc[i] = cla.s.vc[i];
                if (v[i] <= cla.v[i])
                        v[i] += cla.v[i];
        }
}
```

_ZN2clC1EKPl3st2 -> _ZN2clC1EPl3st2

_ZN2cl5elab1ER3st13st2

Primo parametro delle funzioni: puntatore implicito *this**

Registri scratch: RAX, R10, R11, *RCX, RDX, RSI, RDI, R8, R9*.
Registri non scratch: RBP, RBX, R12, R13, R14, R15.
Registri per i parametri in ingresso: RDI, RSI, RDX, RCX, R8, R9
Registri per il valore di ritorno: RDX_RAX

Esercizio di traduzione del 23 settembre 2019

Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

```
struct st1 { char vc[4]; }; struct st2 { char vd[4]; };
class cl {
        long v[4];
        st1 c1; st1 c2;
public:
        cl(char c, st2& s);
        void elab1(st1 s1, st2 s2);
        void stampa()
        {
                for (int i=0; i < 4; i++) cout << v[i] << ' '; cout << "\n";
                for (int i=0; i < 4; i++) cout << c1.vc[i] << ' '; cout << "\n";
                for (int i=0; i < 4; i++) cout << c2.vc[i] << ' '; cout << "\n\n";
        }
};
```

+8	+7	+6	+5	+4	+3	+2	+1	
v[0]								+0
v[1]								+8
v[2]								+16
v[3]								+24
c2.vd[3]	c2.vd[2]	c2.vd[1]	c2.vd[0]	c1.vc[3]	c1.vc[2]	c1.vc[1]	c1.vc[0]	+32

Realizzare in Assembler GCC le funzioni membro seguenti.

```
cl::cl(char c, st2& s2) {
        for (int i = 0; i < 4; i++) {
                c1.vc[i] = c; c2.vc[i] = c++;
                v[i] = s2.vd[i] + c2.vc[i];
        }
}
void cl::elab1(st1 s1, st2 s2) {
        cl cla('a', s2);
        for (int i = 0; i < 4; i++) {
                if (c2.vc[i] <= s1.vc[i]) {
                        c1.vc[i] = i + cla.c2.vc[i];
                        v[i] = i - cla.v[i];
                }
        }
}
```

_ZN2clC1EcR3st2

_ZN2cl5elab1E3st13st2

Primo parametro delle funzioni: puntatore implicito *this**

Registri scratch: RAX, R10, R11, *RCX, RDX, RSI, RDI, R8, R9*.
Registri non scratch: RBP, RBX, R12, R13, R14, R15.
Registri per i parametri in ingresso: RDI, RSI, RDX, RCX, R8, R9
Registri per il valore di ritorno: RDX_RAX

Esercizio di traduzione del 24 luglio 2019

Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

```
struct st {
        long vv2[4];
        char vv1[4];
};
class cl {
        st s;
public:
        cl(char v[]);
        void elab1(st& ss, int d);
        void stampa()
        {
                for (int i = 0; i < 4; i++)
                        cout << (int)s.vv1[i] << ' ';
                cout << '\t';
                for (int i = 0; i < 4; i++)
                        cout << s.vv2[i] << ' ';
                cout << endl;
                cout << endl;
        }
};
```

+8	+7	+6	+5	+4	+3	+2	+1	
s.vv2[0]								+0
s.vv2[1]								+8
s.vv2[2]								+16
s.vv2[3]								+24
X	X	X	X	s.vv1 [3]	s.vv1 [2]	s.vv1 [1]	s.vv1[0]	+32

Realizzare in Assembler GCC le funzioni membro seguenti.

```
cl::cl(char v[])
{
        for (int i = 0; i < 4; i++) {
                s.vv1[i] = s.vv2[i] = v[i];
        }
}
void cl::elab1(int d, st& ss)
{
        for (int i = 0; i < 4; i++) {
                if (d >= ss.vv2[i])
                        s.vv1[i] += ss.vv1[i];
                s.vv2[i] = d + i;
        }
}
```

_ZN2clC1EPv

_ZN2cl5elab1ER2sti

Primo parametro delle funzioni: puntatore implicito *this**

Registri scratch: RAX, R10, R11, *RCX, RDX, RSI, RDI, R8, R9*.
Registri non scratch: RBP, RBX, R12, R13, R14, R15.
Registri per i parametri in ingresso: RDI, RSI, RDX, RCX, R8, R9
Registri per il valore di ritorno: RDX_RAX

Esercizio di traduzione del 3 luglio 2019

Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

```
struct st {
        char vv1[4];
        long vv2[4];
};
class cl {
        st s;
public:
        cl(char v[]);
        void elab1(int d, st& ss);
        void stampa()
        {
                for (int i = 0; i < 4; i++)
                        cout << (int)s.vv1[i] << ' ';
                cout << '\t';
                for (int i = 0; i < 4; i++)
                        cout << s.vv2[i] << ' ';
                cout << endl;
                cout << endl;
        }
};
```

+8	+7	+6	+5	+4	+3	+2	+1	
X	X	X	X	s.vv1[3]	s.vv1[2]	s.vv1[1]	s.vv1[0]	+0
s.vv2[0]								+8
s.vv2[1]								+16
s.vv2[2]								+24
s.vv2[3]								+32

Realizzare in Assembler GCC le funzioni membro seguenti.

```
cl::cl(char c, st2& s2) {
        for (int i = 0; i < 4; i++) {
                c1.vc[i] = c; c2.vc[i] = c++;
                v[i] = s2.vd[i] + c2.vc[i];
        }
}
void cl::elab1(st1 s1, st2 s2) {
        cl cla('a', s2);
        for (int i = 0; i < 4; i++) {
                if (c2.vc[i] <= s1.vc[i]) {
                        c1.vc[i] = i + cla.c2.vc[i];
                        v[i] = i - cla.v[i];
                }
        }
}
```

_ZN2clC1EcR3st2

_ZN2cl5elab1E3st13st2

Primo parametro delle funzioni: puntatore implicito *this**

Registri scratch: RAX, R10, R11, *RCX, RDX, RSI, RDI, R8, R9*.
Registri non scratch: RBP, RBX, R12, R13, R14, R15.
Registri per i parametri in ingresso: RDI, RSI, RDX, RCX, R8, R9
Registri per il valore di ritorno: RDX_RAX

Esercizio di traduzione del 12 giugno 2019

Siano date le seguenti dichiarazioni, contenute nel file `cc.h`:

```
struct st1 {
        char vc[4];
};
class cl {
        long v[4];
        st1 s;
public:
        cl(char c, st1 s2);
        void elab1(st1& s1);
        void stampa()
        {
                for (int i = 0; i < 4 ;i++) cout << s.vc[i] << ' '; cout << endl;
                for (int i = 0; i < 4; i++) cout << v[i] << ' '; cout << endl << endl;
        }
};
```

+8	+7	+6	+5	+4	+3	+2	+1	
v[0]								+0
v[1]								+8
v[2]								+16
v[3]								+24
X	X	X	X	s.vc[3]	s.vc[2]	s.vc[1]	s.vc[0]	+32

Realizzare in Assembler GCC le funzioni membro seguenti.

```
cl::cl(char c, st1 s2)
{
        for (int i = 0; i < 4; i++) {
                s.vc[i] = c;
                v[i] = s2.vc[i] - c;
        }
}
void cl::elab1(st1& s1)
{
        cl cla('x', s1);
        for (int i = 0; i < 4; i++) {
                if (s.vc[i] <= s1.vc[i]) {
                        s.vc[i] = cla.s.vc[i];
                        v[i] = cla.v[i] + i;
                }
        }
}
```

_ZN2clC1Ec3st1

_ZN2cl5elab1ER3st1

Primo parametro delle funzioni: puntatore implicito *this**

Registri scratch: RAX, R10, R11, *RCX, RDX, RSI, RDI, R8, R9*.
Registri non scratch: RBP, RBX, R12, R13, R14, R15.
Registri per i parametri in ingresso: RDI, RSI, RDX, RCX, R8, R9
Registri per il valore di ritorno: RDX_RAX