Si considerino le seguenti definizioni:

```
class A {
public:
    virtual ~A() {}
};

class B: public A {};
class C: public A {};
class D: public B, public C {};
class E: public D {};
class F: public B {};
class G: public F {};

std::string H(A* pa, B* pb) {
    if (dynamic_cast<F*>(pa)) {
        if (dynamic_cast<G*>(pa))
            return "7";
        return "3";
    }

    if (dynamic_cast<D*>(pa)) {
        if (dynamic_cast<E*>(pa))
            return "1";
        return "6";
    }

    if (dynamic_cast<C*>(pb))
        return "4";

    if (typeid(*pa) == typeid(*pb))
        return "8";

    return "0";
}
```

Si consideri inoltre il seguente `main()` incompleto:

```
int main() {
    A a; B b; C c; D d; E e; F f; G g;

    cout << H(..., ...) << H(..., ...) << H(..., ...)
         << H(..., ...) << H(..., ...) << H(..., ...)
         << H(..., ...) << H(..., ...);
}
```

Definire opportunamente le chiamate alla funzione `H()` in questo `main()` usando gli oggetti locali `a`, `b`, `c`, `d`, `e`, `f`, `g` in modo tale che:

1. Il `main()` compili correttamente ed esegua senza provocare errori a run-time
2. Le chiamate ad `H()` siano tutte diverse tra loro
3. L'esecuzione produca in output esattamente la stampa "31415926"