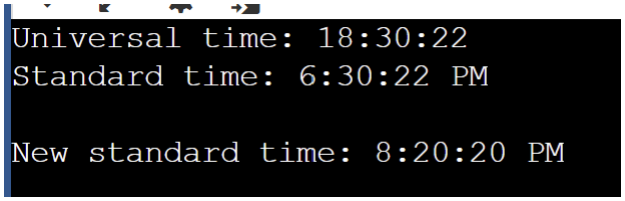
```cpp
#include <iostream>
#include "Time.h"
using namespace std;

int main()
{
    Time t;

    t.setHour(18).setMinute(30).setSecond(22);

    cout << "Universal time: ";
    t.printUniversal();

    cout << "\nStandard time: ";
    t.printStandard();

    cout << "\n\nNew standard time: ";

    t.setTime(20, 20, 20).printStandard();
    cout << endl;

    return 0;
}
```

```cpp
#ifndef TIME_H
#define TIME_H

class Time{
    public:
        Time(int = 0, int = 0, int = 0);

        Time &setTime(int, int, int);
        Time &setHour(int);
        Time &setMinute(int);
        Time &setSecond(int);

        int getHour() const;
        int getMinute() const;
        int getSecond() const;

        void printUniversal() const;
        void printStandard() const;

    private:
        int secPassedFromMid;
};

#endif
```

```cpp
#include <iostream>
#include <iomanip>
#include "Time.h"
using namespace std;

Time::Time(int hr, int min, int sec){
    setTime(hr, min, sec);
}

Time &Time::setTime(int h, int m, int s){
    secPassedFromMid = 0;

    setHour(h);
    setMinute(m);
    setSecond(s);

    return *this;
}

Time &Time::setHour(int h){
    if (h >= 0 and h < 24){
        secPassedFromMid += 3600 * h;
    }

    return *this;
}
```

```cpp
Time &Time::setMinute(int m){
    if (m >= 0 and m < 60){
        secPassedFromMid += 60 * m;
    }

    return *this;
}

Time &Time::setSecond(int s){
    if (s >= 0 and s < 60){
        secPassedFromMid += s;
    }

    return *this;
}

int Time::getHour() const{
    return (secPassedFromMid/3600);
}

int Time::getMinute() const {
    return ( (secPassedFromMid - (secPassedFromMid/3600*3600) ) / 60 );
}

int Time::getSecond() const{
    return (secPassedFromMid % 60);
}
```

```cpp
void Time::printUniversal() const{
    cout << setfill('0') << setw(2) << (secPassedFromMid/3600) << ":"
    << setw(2) << ( (secPassedFromMid - (secPassedFromMid/3600*3600) ) / 60 ) << ":"
    << setw(2) << (secPassedFromMid % 60);
}

void Time::printStandard() const{
    cout << ( ((secPassedFromMid/3600) == 0 or (secPassedFromMid/3600) == 12) ? 12 : (secPassedFromMid/3600) % 12)
    << ":" << setfill('0') << setw(2) << ( (secPassedFromMid - (secPassedFromMid/3600*3600) ) / 60 )
    << ":" << setw(2) << (secPassedFromMid % 60) << ((secPassedFromMid/3600) < 12 ? " AM" : " PM");
}
```


Universal time: 18:30:22
Standard time: 6:30:22 PM
New standard time: 8:20:20 PM