# APÉNDICE 3)
# Como sumar días a una fecha concreta

Para que a una Fecha concreta podamos sumarle una serie de días debemos sobrecargar el operator+ con un método y una función ajena a la clase (la cual podemos declararla amiga de la clase o no, según nos convenga).

```
class Fecha {
  int dia;
  int mes, anio;
public:
  ...
  Fecha operator+(const int &i) const; //f+5
  friend Fecha operator+(const int &i, const Fecha &f);
};
Fecha operator+(const int &i, const Fecha &f); //const por seguridad y & por velocidad
```

Basándonos en el código del operator++ que ya os hemos explicado, podemos intentar codificar el operator+, siguiendo las indicaciones dadas antes.

**Código del operator++ (notación prefija)**

```
Fecha Fecha::operator++() {   //++f
    int dmax, diaMes[] = {0,31,28,31,30,31,30,31,31,30,31,30,31};
    if (this->bisiesto()) //si el año es bisiesto febrero tiene 29 dias
      diaMes[2]=29;
    dmax=diaMes[this->mes];
    this->dia++;
    if (this->dia>dmax) { //si al incrementar dia supera el numero de dias de dicho mes
        this->dia=1;      //pasamos a 1
        this->mes++;      //del mes siguiente
        if (this->mes>12) { //si al incrementar mes pasamos de 12 meses
            this->mes=1;    //pasamos al mes 1
            this->anio++;   //del año siguiente
        }
    }
    return *this; //devolvemos el objeto fecha ya incrementado
}
```

A diferencia del operator++, el operator+ NO DEBE MODIFICAR el objeto fecha que lo invoca, sino que debe devolver un objeto nuevo con el resultado de la suma (cuando sumamos 2 números se devuelve la suma, quedando los 2 números inalterados, de la misma manera cuando a una fecha le sumo unos días, la fecha no cambia sino que se devuelve una fecha nueva resultado de la suma, quedando la fecha original inalterada).

Por tanto, el código del método operator+ será algo parecido a esto:

```
Fecha Fecha::operator+(const int &i) {
    int dmax, diaMes[] = {0,31,28,31,30,31,30,31,31,30,31,30,31};
    if (this->bisiesto()) //si el año es bisiesto febrero tiene 29 dias
      diaMes[2]=29;
    dmax=diaMes[this->mes];

    Fecha suma(dia, mes, anio); //creo fecha local suma igual a la que invoca el método
    suma.dia=dia+i;
    if (suma.dia>dmax) { //si al incrementar dia supera el numero de dias de dicho mes
        suma.dia=suma.dia-dmax;
        suma.mes++;       //del mes siguiente
        if (suma.mes>12) { //si al incrementar mes pasamos de 12 meses
            suma.mes=1;    //pasamos al mes 1
            suma.anio++;   //del año siguiente
        }
    }
    return suma; //devolvemos el objeto fecha suma ya incrementado
}
```

Pero aun así hay otro problema y es que si sumo un número de días muy elevado el código anterior falla, ya que supone que el número de días a sumar no excede de un mes. Por ejemplo si a una fecha le sumo 200 días el mes se debe incrementar varias veces o si sumo 8000 dias el año se debe incrementar varias veces y hay que tener en cuenta si los años posteriores son bisiestos...

Por tanto el código anterior hay que modificarlo para tener en cuenta esto: debo hacer un bucle while para ir restando los días que tienen los meses siguientes hasta agotar todos los días que sumo, de forma que el código correcto sería el siguiente:

```
Fecha Fecha::operator+(const int &i) const {
    int dmax, diaMes[] = {0,31,28,31,30,31,30,31,31,30,31,30,31};
    if (this->bisiesto()) //si el año es bisiesto febrero tiene 29 dias
      diaMes[2]=29;
    dmax=diaMes[this->mes];

    Fecha suma(dia, mes, anio); //creo fecha local suma igual a la que invoca el método
    suma.dia=dia+i;
    while (suma.dia>dmax) { //mientras dia supere el numero de dias de dicho mes
        suma.dia=suma.dia-dmax;
        suma.mes++;       //del mes siguiente
        if (suma.mes>12) { //si al incrementar mes pasamos de 12 meses
            suma.mes=1;    //pasamos al mes 1
            suma.anio++;   //del año siguiente
          //como cambiamos de año debo ver cuanto tiene febrero (28 o 29)
            if (suma.bisiesto()) //si el año es bisiesto febrero tiene 29 dias
              diaMes[2]=29;
            else //si no, lo dejo en 28
              diaMes[2]=28;
        }
        dmax=diaMes[suma.mes]; //como he cambiado de mes compruebo
    }                          //cuantos dias tiene ese mes
    return suma; //devolvemos el objeto fecha suma ya incrementado
}
```

Como podemos ver, la solución es bastante compleja. No obstante hay una solución más trivial si tienes en cuenta el consejo 4 del enunciado de la práctica que dice que:

**Consejos/Ayuda:**

1. ...
2. ...
3. ...
4. Sumar una serie de días a una determinada fecha es equivalente a incrementar dicha fecha tantas veces como días queremos sumar… (¿lo pillas?)

Por tanto, podemos tener una implementación breve, simple y trivial del método operator+ con tan solo crear un bucle for que itere tantas veces como días debemos incrementar (en cada iteración sólo tenemos que llamar al operator++ que ya tenemos):

```
Fecha Fecha::operator+(const int &i) const {
    Fecha suma(dia, mes, anio); //creo fecha local suma igual a la que invoca el método
    for (int n=1; n<=i; n++)
      ++suma; //llamo al operator++ (notacion prefija) que ya tenemos hecha
    return suma; //devolvemos el objeto fecha suma ya incrementado
};
```

Esta implementación da el mismo resultado que la anterior y es muy breve y fácil de hacer (aunque menos eficiente).