

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM



MÔN LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
BÀI TẬP THỰC HÀNH 3

GVHD: Nguyễn Ngọc Quý

Sinh viên thực hiện: Nguyễn Thanh Tùng

[illegible]

(Ký tên và ghi rõ họ tên)

Mục Lục

Bài tập 1: Xây dựng lớp phân số: 12

| | |
|---|----|
| 1.1 – Phương thức của class PhanSo | 14 |
| 1.1.1 – Phương thức PhanSo(): | 14 |
| 1.1.2 – Phương thức PhanSo(int Tu, int Mau): | 14 |
| 1.1.3 – Phương thức RutGon(int &Tu, int &Mau): | 15 |
| 1.1.4 – Phương thức operator+(PhanSo another):..... | 16 |
| 1.1.5 – Phương thức operator-(PhanSo another):..... | 18 |
| 1.1.6 – Phương thức operator*(PhanSo another): | 20 |
| 1.1.7 – Phương thức operator/(PhanSo another): | 22 |
| 1.1.8 – Phương thức operator==(PhanSo another):..... | 24 |
| 1.1.9 – Phương thức operator!=(PhanSo another): | 24 |
| 1.1.10 – Phương thức operator>=(PhanSo another): | 25 |
| 1.1.11 – Phương thức operator<=(PhanSo another): | 25 |
| 1.1.12 – Phương thức operator>(PhanSo another):..... | 26 |
| 1.1.13 – Phương thức operator<(PhanSo another):..... | 27 |
| 1.1.14 – Phương thức operator>>(istream &in, PhanSo &ps):..... | 27 |
| 1.1.15 – Phương thức operator<<(ostream &out, PhanSo &ps):..... | 28 |

Bài tập 2: Xây dựng lớp số phức: 29

| | |
|---|----|
| 2.1 – Phương thức của class SoPhuc: | 31 |
| 2.1.1 – Phương thức SoPhuc()..... | 31 |
| 2.1.2 – Phương thức SoPhuc(int thuc, int ao) | 31 |
| 2.1.3 – Phương thức operator+(SoPhuc another)..... | 32 |
| 2.1.4 – Phương thức operator-(SoPhuc another)..... | 33 |
| 2.1.5 – Phương thức operator*(SoPhuc another) | 33 |
| 2.1.6 – Phương thức operator/(SoPhuc another) | 34 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| | |
|---|-----------|
| 2.1.7 – Phương thức operator==(SoPhuc another)..... | 35 |
| 2.1.8 – Phương thức operator!=(SoPhuc another)..... | 36 |
| 2.1.9 – Phương thức operator>>(istream &in, SoPhuc &another) | 36 |
| 2.1.10 – Phương thức operator<<(ostream &out, SoPhuc &another)..... | 37 |
| Bài tập 3. Xây dựng lớp thời gian..... | 37 |
| 3.1 – Phương thức của class ThoiGian: | 40 |
| 3.1.1 – Phương thức ThoiGian(): | 40 |
| 3.1.2 – Phương thức ThoiGian(int Gio, int Phut, int Giay): | 41 |
| 3.1.3 – Phương thức TinhGiay(): | 41 |
| 3.1.4 – Phương thức TinhLaiGio():..... | 42 |
| 3.1.5 – Phương thức operator+(int Giay): | 43 |
| 3.1.6 – Phương thức operator-(int Giay): | 45 |
| 3.1.7 – Phương thức operator+(ThoiGian a): | 47 |
| 3.1.8 – Phương thức operator-(ThoiGian a): | 49 |
| 3.1.9 – Phương thức operator++(int):..... | 51 |
| 3.1.10 – Phương thức operator--(int): | 54 |
| 3.1.11 – Phương thức operator==(ThoiGian another): | 57 |
| 3.1.12 - Phương thức operator!=(ThoiGian another): | 57 |
| 3.1.13 - Phương thức operator>=(ThoiGian another): | 57 |
| 3.1.14 - Phương thức operator<=(ThoiGian another): | 58 |
| 3.1.15 - Phương thức operator>(ThoiGian another): | 58 |
| 3.1.16 - Phương thức operator<(ThoiGian another): | 59 |
| 3.1.17 - Phương thức operator>>(istream &in, ThoiGian &b): | 59 |
| 3.1.18 - Phương thức operator<<(ostream &out, ThoiGian &b): | 60 |
| Bài tập 4. Xây dựng lớp ngày tháng năm | 61 |
| 4.1 - Phương thức của class NgayThangNam: | 63 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| | |
|---|----|
| 4.1.1 - Phương thức NgayThangNam():..... | 63 |
| 4.1.2 - Phương thức NgayThangNam(int Nam, int Thang, int Ngay): | 64 |
| 4.1.3 - Phương thức TinhNgay(): | 64 |
| 4.1.4 - Phương thức operator+(int ngay):..... | 66 |
| 4.1.5 - Phương thức operator-(int ngay):..... | 67 |
| 4.1.6 - Phương thức toDays() const:..... | 69 |
| 4.1.7 - Phương thức operator-(const NgayThangNam &a) const: | 71 |
| 4.1.8 - Phương thức operator++(int): | 72 |
| 4.1.9 - Phương thức operator--(int): | 73 |
| 4.1.10 - Phương thức operator==(NgayThangNam another):..... | 75 |
| 4.1.11 - Phương thức operator!=(NgayThangNam another):..... | 76 |
| 4.1.12 - Phương thức operator>=(NgayThangNam another):..... | 76 |
| 4.1.13 - Phương thức operator<=(NgayThangNam another):..... | 78 |
| 4.1.14 - Phương thức operator>(NgayThangNam another): | 79 |
| 4.1.15 - Phương thức operator<(NgayThangNam another): | 81 |
| 4.1.16 - Phương thức operator<<(ostream &out, NgayThangNam &ntn): | 82 |
| 4.1.17 - Phương thức operator>>(ostream &out, NgayThangNam &ntn): | 82 |

DANH MỤC BẢNG

| | |
|---|----|
| Bảng 1. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo.. | 14 |
| Bảng 2. Nội dung của phương thức PhanSo() | 14 |
| Bảng 3. Nội dung của phương thức PhanSo(int Tu, int Mau)..... | 15 |
| Bảng 4. Nội dung của phương thức RutGon(int &Tu, int &Mau)..... | 16 |
| Bảng 5. Ví dụ về Input/Output của phương thức RutGon(int &Tu, int &Mau) | 16 |
| Bảng 6. Nội dung của phương thức operator+(PhanSo another) | 18 |
| Bảng 7. Ví dụ Input/Output của phương thức operator+(PhanSo another) | 18 |
| Bảng 8. Nội dung của phương thức operator-(PhanSo another) | 20 |
| Bảng 9. Ví dụ Input/Output của phương thức operator-(PhanSo another) | 20 |
| Bảng 10. Nội dung phương thức operator*(PhanSo another) | 22 |
| Bảng 11. Ví dụ về Input/Output của phương thức operator*(PhanSo another) | 22 |
| Bảng 12. Nội dung phương thức operator/(PhanSo another) | 24 |
| Bảng 13. Ví dụ về Input/Output của phương thức operator/(PhanSo another) | 24 |
| Bảng 14. Nội dung của phương thức operator==(PhanSo another) | 24 |
| Bảng 15. Nội dung của phương thức operator!=(PhanSo another) | 25 |
| Bảng 16. Nội dung của phương thức operator>=(PhanSo another) | 25 |
| Bảng 17. Nội dung của phương thức operator<=(PhanSo another) | 26 |
| Bảng 18. Nội dung của phương thức operator>(PhanSo another) | 27 |
| Bảng 19. Nội dung của phương thức operator<(PhanSo another) | 27 |
| Bảng 20. Nội dung của phương thức operator>>(istream &in, PhanSo &ps) | 28 |
| Bảng 21. Nội dung của phương thức operator<<(istream &in, PhanSo &ps) | 28 |
| Bảng 22. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc | 31 |
| Bảng 23. Nội dung của phương thức SoPhuc() | 31 |
| Bảng 24. Nội dung của phương thức SoPhuc(int thuc, int ao)..... | 32 |
| Bảng 25. Nội dung của phương thức operator+(SoPhuc another) | 32 |
| Bảng 26. Ví dụ Input/Output của phương thức operator+(SoPhuc another) | 32 |
| Bảng 27. Nội dung của phương thức operator-(SoPhuc another) | 33 |
| Bảng 28. Ví dụ về Input/Output của phương thức operator-(SoPhuc another)..... | 33 |
| Bảng 29. Nội dung của phương thức operator*(SoPhuc another) | 34 |
| Bảng 30. Ví dụ Input/Output của phương thức operator*(SoPhuc another)..... | 34 |
| Bảng 31. Nội dung của phương thức operator/(SoPhuc another) | 35 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| | |
|---|----|
| Bảng 32. Ví dụ Input/Output của phương thức operator/(SoPhuc another)..... | 35 |
| Bảng 33. Nội dung của phương thức operator==(SoPhuc another)..... | 36 |
| Bảng 34. Nội dung của phương thức operator!=(SoPhuc another)..... | 36 |
| Bảng 35. Nội dung phương thức operator>>(istream &in, SoPhuc &another) | 37 |
| Bảng 36. Nội dung của phương thức operator<<(ostream &out, SoPhuc &another)... | 37 |
| Bảng 37. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian | 40 |
| Bảng 38. Nội dung của phương thức ThoiGian()..... | 41 |
| Bảng 39. Nội dung của phương thức ThoiGian(int Gio, int Phut, int Giay)..... | 41 |
| Bảng 40. Nội dung của phương thức TinhGiay()..... | 42 |
| Bảng 41. Ví dụ Input/Output của phương thức TinhGiay() | 42 |
| Bảng 42. Nội dung của phương thức TinhLaiGio(). | 43 |
| Bảng 43. Ví dụ Input/Output của phương thức TinhLaiGio(). | 43 |
| Bảng 44. Nội dung của phương thức operator+(int Giay) | 45 |
| Bảng 45. Ví dụ Input/Output của phương thức operator+(int Giay)..... | 45 |
| Bảng 46. Nội dung của phương thức operator-(int Giay) | 47 |
| Bảng 47. Ví dụ Input/Output của phương thức operator-(int Giay)..... | 47 |
| Bảng 48. Nội dung của phương thức operator+(ThoiGian a) | 49 |
| Bảng 49. Ví dụ Input/Output của phương thức operator+(ThoiGian a)..... | 49 |
| Bảng 50. Nội dung của phương thức operator-(ThoiGian a) | 51 |
| Bảng 51. Ví dụ Input/Output của phương thức operator-(ThoiGian a) | 51 |
| Bảng 52. Nội dung phương thức operator++(int)..... | 53 |
| Bảng 53. Ví dụ Input/Output của phương thức operator++(int) | 54 |
| Bảng 54. Nội dung của phương thức operator--(int)..... | 56 |
| Bảng 55. Ví dụ Input/Output của phương thức operator--(int) | 57 |
| Bảng 56. Nội dung của phương thức operator==(ThoiGian another)..... | 57 |
| Bảng 57. Nội dung của phương thức operator!=(ThoiGian another)..... | 57 |
| Bảng 58. Nội dung của phương thức operator>=(ThoiGian another)..... | 58 |
| Bảng 59. Nội dung của phương thức operator>=(ThoiGian another)..... | 58 |
| Bảng 60. Nội dung của phương thức operator>(ThoiGian another) | 59 |
| Bảng 61. Nội dung của phương thức operator<(ThoiGian another) | 59 |
| Bảng 62. Nội dung của phương thức operator>>(istream &in, ThoiGian &b)..... | 60 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| | |
|--|----|
| Bảng 63. Nội dung của phương thức operator<<(ostream &out, ThoiGian &b)..... | 61 |
| Bảng 64. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam..... | 63 |
| Bảng 65. Nội dung của phương thức NgayThangNam() | 64 |
| Bảng 66. Nội dung của phương thức NgayThangNam(int Nam, int Thang, int Ngay) | 64 |
| Bảng 67. Nội dung của phương thức TinhNgay()..... | 65 |
| Bảng 68. Ví dụ Input/Output của phương thức TinhNgay() | 66 |
| Bảng 69. Nội dung của phương thức operator+(int ngay) | 67 |
| Bảng 70. Ví dụ Input/Output của phương thức operator+(int ngay)..... | 67 |
| Bảng 71. Nội dung của phương thức operator-(int ngay) | 69 |
| Bảng 72. Nội dung của phương thức operator-(int ngay) | 69 |
| Bảng 73. Nội dung của phương thức toDays() const | 71 |
| Bảng 74. Ví dụ Input/Output của phương thức toDays() const..... | 71 |
| Bảng 75. Nội dung của phương thức operator-(const NgayThangNam &a) const..... | 72 |
| Bảng 76. Ví dụ Input/Output của phương thức operator-(const NgayThangNam &a) const..... | 72 |
| Bảng 77. Nội dung của phương thức operator++(int) | 73 |
| Bảng 78. Ví dụ Input/Output của phương thức operator++(int) | 73 |
| Bảng 79. Nội dung của phương thức operator--(int)..... | 75 |
| Bảng 80. Ví dụ Input/Output của phương thức operator--(int) | 75 |
| Bảng 81. Nội dung của phương thức operator==(NgayThangNam another) | 76 |
| Bảng 82. Nội dung của phương thức operator!=(NgayThangNam another) | 76 |
| Bảng 83. Nội dung của phương thức operator>=(NgayThangNam another) | 78 |
| Bảng 84. Nội dung của phương thức operator<=(NgayThangNam another) | 79 |
| Bảng 85. Nội dung của phương thức operator>(NgayThangNam another)..... | 81 |
| Bảng 86. Nội dung của phương thức operator<(NgayThangNam another)..... | 82 |
| Bảng 87. Nội dung của phương thức operator<<(ostream &out, NgayThangNam &ntn)..... | 82 |
| Bảng 88. Nội dung của phương thức operator>>(ostream &out, NgayThangNam &ntn)..... | 83 |

DANH MỤC HÌNH ẢNH

| | |
|--|----|
| Hình 1. Class diagram của lớp PhanSo | 12 |
| Hình 2. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo .. | 13 |
| Hình 3. Nội dung của phương thức PhanSo() | 14 |
| Hình 4. Nội dung của phương thức PhanSo(int Tu, int Mau) | 15 |
| Hình 5. Nội dung của phương thức RutGon(int &Tu, int &Mau) | 15 |
| Hình 6. Nội dung của phương thức operator+(PhanSo another)..... | 17 |
| Hình 7. Nội dung của phương thức operator-(PhanSo another)..... | 19 |
| Hình 8. Nội dung phương thức operator*(PhanSo another) | 21 |
| Hình 9. Nội dung phương thức operator/(PhanSo another) | 23 |
| Hình 10. Nội dung của phương thức operator==(PhanSo another) | 24 |
| Hình 11. Nội dung của phương thức operator!=(PhanSo another) | 25 |
| Hình 12. Nội dung của phương thức operator>=(PhanSo another) | 25 |
| Hình 13. Nội dung của phương thức operator<=(PhanSo another) | 26 |
| Hình 14. Nội dung của phương thức operator>(PhanSo another)..... | 26 |
| Hình 15. Nội dung của phương thức operator<(PhanSo another)..... | 27 |
| Hình 16. Nội dung của phương thức operator>>(istream &in, PhanSo &ps)..... | 28 |
| Hình 17. Nội dung của phương thức operator<<(istream &in, PhanSo &ps)..... | 28 |
| Hình 18. Class diagram của lớp SoPhuc | 29 |
| Hình 19. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc | 30 |
| Hình 20. Nội dung của phương thức SoPhuc() | 31 |
| Hình 21. Nội dung của phương thức SoPhuc(int thuc, int ao) | 31 |
| Hình 22. Nội dung của phương thức operator+(SoPhuc another)..... | 32 |
| Hình 23. Nội dung của phương thức operator-(SoPhuc another)..... | 33 |
| Hình 24. Nội dung của phương thức operator*(SoPhuc another)..... | 34 |
| Hình 25. Nội dung của phương thức operator/(SoPhuc another)..... | 35 |
| Hình 26. Nội dung của phương thức operator==(SoPhuc another) | 36 |
| Hình 27. Nội dung của phương thức operator!=(SoPhuc another) | 36 |
| Hình 28. Nội dung phương thức operator>>(istream &in, SoPhuc &another)..... | 36 |
| Hình 29. Nội dung của phương thức operator<<(ostream &out, SoPhuc &another) ... | 37 |
| Hình 30. Class diagram của lớp ThoiGian | 38 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| | |
|---|----|
| Hình 31. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian | 39 |
| Hình 32. Nội dung của phương thức ThoiGian() | 40 |
| Hình 33. Nội dung của phương thức ThoiGian(int Gio, int Phut, int Giay) | 41 |
| Hình 34. Nội dung của phương thức TinhGiay() | 42 |
| Hình 35. Nội dung của phương thức TinhLaiGio()..... | 42 |
| Hình 36. Nội dung của phương thức operator+(int Giay)..... | 44 |
| Hình 37. Nội dung của phương thức operator-(int Giay)..... | 46 |
| Hình 38. Nội dung của phương thức operator+(ThoiGian a)..... | 48 |
| Hình 39. Nội dung của phương thức operator-(ThoiGian a)..... | 50 |
| Hình 40. Nội dung phương thức operator++(int) | 52 |
| Hình 41. Nội dung của phương thức operator--(int) | 55 |
| Hình 42. Nội dung của phương thức operator==(ThoiGian another) | 57 |
| Hình 43. Nội dung của phương thức operator!=(ThoiGian another) | 57 |
| Hình 44. Nội dung của phương thức operator>=(ThoiGian another) | 58 |
| Hình 45. Nội dung của phương thức operator>=(ThoiGian another) | 58 |
| Hình 46. Nội dung của phương thức operator>(ThoiGian another) | 59 |
| Hình 47. Nội dung của phương thức operator<(ThoiGian another) | 59 |
| Hình 48. Nội dung của phương thức operator>>(istream &in, ThoiGian &b) | 60 |
| Hình 49. Nội dung của phương thức operator<<(ostream &out, ThoiGian &b) | 60 |
| Hình 50. Class diagram của lớp NgayThangNam..... | 61 |
| Hình 51. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam..... | 62 |
| Hình 52. Nội dung của phương thức NgayThangNam()..... | 63 |
| Hình 53. Nội dung của phương thức NgayThangNam(int Nam, int Thang, int Ngay) | 64 |
| Hình 54. Nội dung của phương thức TinhNgay() | 65 |
| Hình 55. Nội dung của phương thức operator+(int ngay)..... | 66 |
| Hình 56. Nội dung của phương thức operator-(int ngay)..... | 68 |
| Hình 57. Nội dung của phương thức toDays() const..... | 70 |
| Hình 58. Nội dung của phương thức operator-(const NgayThangNam &a) const | 71 |
| Hình 59. Nội dung của phương thức operator++(int) | 72 |
| Hình 60. Nội dung của phương thức operator--(int) | 74 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| | |
|---|----|
| Hình 61. Nội dung của phương thức operator==(NgayThangNam another) | 75 |
| Hình 62. Nội dung của phương thức operator!=(NgayThangNam another) | 76 |
| Hình 63. Nội dung của phương thức operator>=(NgayThangNam another) | 77 |
| Hình 64. Nội dung của phương thức operator<=(NgayThangNam another) | 78 |
| Hình 65. Nội dung của phương thức operator>(NgayThangNam another) | 80 |
| Hình 66. Nội dung của phương thức operator<(NgayThangNam another) | 81 |
| Hình 67. Nội dung của phương thức operator<<(ostream &out, NgayThangNam &n) | 82 |
| Hình 68. Nội dung của phương thức operator>>(ostream &out, NgayThangNam &n) | 83 |

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

NỘI DUNG BÀI LÀM

Bài tập 1: Xây dựng lớp phân số:

- Thuộc tính: iTu, iMau
- Phương thức: PhanSo(), PhanSo(int Tu, int Mau)
- Thực hiện các phương thức operator: +, -, *, /, ==, !=, >=, <=, >, <, <<

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm main()

Class diagram của lớp PhanSo:

| PhanSo |
|--|
| - iTu: int - iMau: int |
| + PhanSo() + PhanSo(int Tu, int Mau) + void RutGon(int Tu, int Mau): void + operator+(PhanSo another): PhanSo + operator-(PhanSo another): PhanSo + operator*(PhanSo another): PhanSo + operator/(PhanSo another): PhanSo + operator==(PhanSo another): bool + operator!=(PhanSo another): bool + operator>=(PhanSo another): bool + operator<=(PhanSo another): bool + operator>(PhanSo another): bool + operator<(PhanSo another): bool + operator>>(istream &in, PhanSo &ps): friend + operator<<(ostream &out, PhanSo &ps): friend |

Hình 1. Class diagram của lớp PhanSo

Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
1  #include <iostream>      You, 3 weeks ago • First commit
2
3  using namespace std;
4
5  class PhanSo
6  {
7  private:
8      int iTu, iMau;
9
10 public:
11     PhanSo();
12     PhanSo(int Tu, int Mau);
13     void RutGon(int &Tu, int &Mau);
14     PhanSo operator+(PhanSo another);
15     PhanSo operator-(PhanSo another);
16     PhanSo operator*(PhanSo another);
17     PhanSo operator/(PhanSo another);
18     bool operator==(PhanSo another);
19     bool operator!=(PhanSo another);
20     bool operator>=(PhanSo another);
21     bool operator<=(PhanSo another);
22     bool operator>(PhanSo another);
23     bool operator<(PhanSo another);
24     friend istream &operator>>(istream &in, PhanSo &ps);
25     friend ostream &operator<<(ostream &out, PhanSo &ps);
26 };
```

Hình 2. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo

```
#include <iostream>

using namespace std;

class PhanSo
{
private:
    int iTu, iMau;

public:
    PhanSo();
    PhanSo(int Tu, int Mau);
    void RutGon(int &Tu, int &Mau);
```

```
PhanSo operator+(PhanSo another);
PhanSo operator-(PhanSo another);
PhanSo operator*(PhanSo another);
PhanSo operator/(PhanSo another);
bool operator==(PhanSo another);
bool operator!=(PhanSo another);
bool operator>=(PhanSo another);
bool operator<=(PhanSo another);
bool operator>(PhanSo another);
bool operator<(PhanSo another);
friend istream &operator>>(istream &in, PhanSo &ps);
friend ostream &operator<<(ostream &out, PhanSo &ps);
};
```

Bảng 1. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo

1.1 – Phương thức của class PhanSo

1.1.1 – Phương thức PhanSo():

- Nội dung: Gán giá trị ban đầu của iTu, iMau = 1.

```
41 PhanSo::PhanSo()
42 {
43     iTu = 1;
44     iMau = 1;
45 }
```

Hình 3. Nội dung của phương thức PhanSo()

```
PhanSo::PhanSo()
{
    iTu = 1;
    iMau = 1;
}
```

Bảng 2. Nội dung của phương thức PhanSo()

1.1.2 – Phương thức PhanSo(int Tu, int Mau):

- Nội dung: Gán giá trị nhập vào biến iTu, iMau.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
62 PhanSo::PhanSo(int Tu, int Mau)
63 {
64     iTu = Tu;
65     iMau = Mau;
66 }
```

Hình 4. Nội dung của phương thức PhanSo(int Tu, int Mau)

```
PhanSo::PhanSo(int Tu, int Mau)
{
    iTu = Tu;
    iMau = Mau;
}
```

Bảng 3. Nội dung của phương thức PhanSo(int Tu, int Mau)

1.1.3 – Phương thức RutGon(int &Tu, int &Mau):

- Input (đầu vào): Giá trị Tử, Mẫu muốn rút gọn.
- Output (đầu ra): Giá trị Tử, Mẫu sau khi rút gọn.
- Hướng giải quyết: Tìm ước chung lớn nhất của Tử và Mẫu, sau đó lấy Tử và Mẫu chia cho ước chung lớn nhất.

```
28 void PhanSo::RutGon(int &Tu, int &Mau)
29 {
30     int x = Tu, y = Mau;
31     while (y != 0)
32     {
33         int temp = y;
34         y = x % y;
35         x = temp;
36     }
37     Tu /= x;
38     Mau /= x;
39 }
```

Hình 5. Nội dung của phương thức RutGon(int &Tu, int &Mau)

```
void PhanSo::RutGon(int &Tu, int &Mau)
{
    int x = Tu, y = Mau;
```

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
while (y != 0)
{
    int temp = y;
    y = x % y;
    x = temp;
}
Tu /= x;
Mau /= x;
}
```

Bảng 4. Nội dung của phương thức RutGon(int &Tu, int &Mau)

| |
|-------------|
| Input: 2 4 |
| Output: 1 2 |

Bảng 5. Ví dụ về Input/Output của phương thức RutGon(int &Tu, int &Mau)

1.1.4 – Phương thức operator+(PhanSo another):

- Input (đầu vào): Phân số muốn cộng.
- Output (đầu ra): Tổng 2 phân số sau khi cộng.
- Hướng giải quyết: Quy đồng, sau đó cộng 2 phân số => rút gọn => in ra phân số.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
68  PhanSo PhanSo::operator+(PhanSo another)
69  {
70      cout << "Tong cua 2 phan so la: ";
71      PhanSo kq;
72      kq.iTu = iTu * another.iMau + iMau * another.iTu;
73      kq.iMau = iMau * another.iMau;
74      if (kq.iTu == 0)
75      {
76          cout << 0 << endl;
77      }
78      else if (kq.iTu == kq.iMau)
79      {
80          cout << 1 << endl;
81      }
82      else if (kq.iTu % kq.iMau == 0)
83      {
84          cout << kq.iTu / kq.iMau << endl;
85      }
86      else
87      {
88          RutGon(kq.iTu, kq.iMau);
89          cout << kq.iTu << '/' << kq.iMau << endl;
90      }
91      return kq;
92  }
```

Hình 6. Nội dung của phương thức `operator+(PhanSo another)`

```
PhanSo PhanSo::operator+(PhanSo another)
{
    cout << "Tong cua 2 phan so la: ";
    PhanSo kq;
    kq.iTu = iTu * another.iMau + iMau * another.iTu;
    kq.iMau = iMau * another.iMau;
    if (kq.iTu == 0)
    {
        cout << 0 << endl;
    }
    else if (kq.iTu == kq.iMau)
    {
        cout << 1 << endl;
```

```

    }
    else if (kq.iTu % kq.iMau == 0)
    {
        cout << kq.iTu / kq.iMau << endl;
    }
    else
    {
        RutGon(kq.iTu, kq.iMau);
        cout << kq.iTu << '/' << kq.iMau << endl;
    }
    return kq;
}

```

Bảng 6. Nội dung của phương thức operator+(PhanSo another)

| |
|----------------------------------|
| Input: 1 2 |
| //Giả sử phân số hiện tại là 1 1 |
| Output: 3/2 |

*Bảng 7. Ví dụ Input/Output của phương thức operator+(PhanSo another)***1.1.5 – Phương thức operator-(PhanSo another):**

- Input (đầu vào): Phân số muốn trừ.
Output (đầu ra): Hiệu 2 phân số.
- Hướng giải quyết: Quy đồng mẫu số => trừ 2 phân số => rút gọn => in ra kết quả

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
94 PhanSo PhanSo::operator-(PhanSo another)
95 {
96     cout << "Hieu cua 2 phan so la: ";
97     PhanSo kq;
98     kq.iTu = iTu * another.iMau - iMau * another.iTu;
99     kq.iMau = iMau * another.iMau;
100     if (kq.iTu == 0)
101     {
102         cout << 0 << endl;
103     }
104     else if (kq.iTu == kq.iMau)
105     {
106         cout << 1 << endl;
107     }
108     else if (kq.iTu % kq.iMau == 0)
109     {
110         cout << kq.iTu / kq.iMau << endl;
111     }
112     else
113     {
114         RutGon(kq.iTu, kq.iMau);
115         cout << kq.iTu << '/' << kq.iMau << endl;
116     }
117     return kq;
118 }
```

Hình 7. Nội dung của phương thức operator-(PhanSo another)

```
PhanSo PhanSo::operator-(PhanSo another)
{
    cout << "Hieu cua 2 phan so la: ";
    PhanSo kq;
    kq.iTu = iTu * another.iMau - iMau * another.iTu;
    kq.iMau = iMau * another.iMau;
    if (kq.iTu == 0)
    {
        cout << 0 << endl;
    }
    else if (kq.iTu == kq.iMau)
    {
        cout << 1 << endl;
```

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
}  
else if (kq.iTu % kq.iMau == 0)  
{  
    cout << kq.iTu / kq.iMau << endl;  
}  
else  
{  
    RutGon(kq.iTu, kq.iMau);  
    cout << kq.iTu << '/' << kq.iMau << endl;  
}  
return kq;  
}
```

Bảng 8. Nội dung của phương thức operator-(PhanSo another)

Input: 1 2

//Giả sử phân số bị trừ là 3/2

Output: 1

Bảng 9. Ví dụ Input/Output của phương thức operator-(PhanSo another)

1.1.6 – Phương thức operator*(PhanSo another):

- Input (đầu vào): Phân số muốn nhân.
Output (đầu ra): Tích 2 phân số.
- Hướng giải quyết: Nhân 2 phân số => rút gọn => in ra màn hình.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
120 PhanSo PhanSo::operator*(PhanSo another)
121 {
122     cout << "Tich cua 2 phan so la: ";
123     PhanSo kq;
124     kq.iTu = iTu * another.iTu;
125     kq.iMau = iMau * another.iMau;
126     if (kq.iTu == 0)
127     {
128         cout << 0 << endl;
129     }
130     else if (kq.iTu == kq.iMau)
131     {
132         cout << 1 << endl;
133     }
134     else if (kq.iTu % kq.iMau == 0)
135     {
136         cout << kq.iTu / kq.iMau << endl;
137     }
138     else
139     {
140         RutGon(kq.iTu, kq.iMau);
141         cout << kq.iTu << '/' << kq.iMau << endl;
142     }
143     return kq;
144 }
```

Hình 8. Nội dung phương thức operator*(PhanSo another)

```
PhanSo PhanSo::operator*(PhanSo another)
{
    cout << "Tich cua 2 phan so la: ";
    PhanSo kq;
    kq.iTu = iTu * another.iTu;
    kq.iMau = iMau * another.iMau;
    if (kq.iTu == 0)
    {
        cout << 0 << endl;
    }
    else if (kq.iTu == kq.iMau)
    {
        cout << 1 << endl;
    }
    else if (kq.iTu % kq.iMau == 0)
    {
        cout << kq.iTu / kq.iMau << endl;
    }
}
```

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
}  
else  
{  
    RutGon(kq.iTu, kq.iMau);  
    cout << kq.iTu << '/' << kq.iMau << endl;  
}  
return kq;  
}
```

Bảng 10. Nội dung phương thức operator(PhanSo another)*

Input: 1 2

//Giả sử phân số hiện tại là 3/4

Output: 3/8

Bảng 11. Ví dụ về Input/Output của phương thức operator(PhanSo another)*

1.1.7 – Phương thức operator/(PhanSo another):

- Input (đầu vào): Phân số bị chia.
Output (đầu ra): Thương 2 phân số.
- Hướng giải quyết: Chia 2 phân số => rút gọn => in ra màn hình.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
146 PhanSo PhanSo::operator/(PhanSo another)
147 {
148     cout << "Thuong cua 2 phan so la: ";
149     PhanSo kq;
150     kq.iTu = iTu * another.iMau;
151     kq.iMau = iMau * another.iTu;
152     if (kq.iTu == 0)
153     {
154         cout << 0 << endl;
155     }
156     else if (kq.iTu == kq.iMau)
157     {
158         cout << 1 << endl;
159     }
160     else if (kq.iTu % kq.iMau == 0)
161     {
162         cout << kq.iTu / kq.iMau << endl;
163     }
164     else
165     {
166         RutGon(kq.iTu, kq.iMau);
167         cout << kq.iTu << '/' << kq.iMau << endl;
168     }
169     return kq;
170 }
```

Hình 9. Nội dung phương thức operator/(PhanSo another)

```
PhanSo PhanSo::operator/(PhanSo another)
{
    cout << "Thuong cua 2 phan so la: ";
    PhanSo kq;
    kq.iTu = iTu * another.iMau;
    kq.iMau = iMau * another.iTu;
    if (kq.iTu == 0)
    {
        cout << 0 << endl;
    }
    else if (kq.iTu == kq.iMau)
    {
        cout << 1 << endl;
    }
    else if (kq.iTu % kq.iMau == 0)
    {
```

```

        cout << kq.iTu / kq.iMau << endl;
    }
    else
    {
        RutGon(kq.iTu, kq.iMau);
        cout << kq.iTu << '/' << kq.iMau << endl;
    }
    return kq;
}

```

Bảng 12. Nội dung phương thức operator/(PhanSo another)

Input: 2 3

//Giả sử số bị chia là 3/2

Output: 1

Bảng 13. Ví dụ về Input/Output của phương thức operator/(PhanSo another)

1.1.8 – Phương thức operator==(PhanSo another):

- Nội dung: Phép so sánh bằng, nếu phân số thứ nhất bằng phân số thứ 2 thì kết quả được trả về.

```

172  bool PhanSo::operator==(PhanSo another)
173  {
174      |      return iTu == another.iTu && iMau == another.iMau;
175  }

```

Hình 10. Nội dung của phương thức operator==(PhanSo another)

```

bool PhanSo::operator==(PhanSo another)
{
    return iTu == another.iTu && iMau == another.iMau;
}

```

Bảng 14. Nội dung của phương thức operator==(PhanSo another)

1.1.9 – Phương thức operator!=(PhanSo another):

- Nội dung: Phép so sánh không bằng, nếu phân số thứ nhất không bằng phân số thứ 2 thì kết quả được trả về.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
177     bool PhanSo::operator!=(PhanSo another)
178     {
179         return !(*this == another);
180     }
```

Hình 11. Nội dung của phương thức operator!=(PhanSo another)

```
bool PhanSo::operator!=(PhanSo another)
{
    return !(*this == another);
}
```

Bảng 15. Nội dung của phương thức operator!=(PhanSo another)

1.1.10 – Phương thức operator>=(PhanSo another):

- Nội dung: Phép so sánh lớn hơn hoặc bằng, nếu phân số thứ nhất lớn hơn hoặc bằng phân số thứ 2 thì kết quả được trả về.

```
182     bool PhanSo::operator>=(PhanSo another)
183     {
184         PhanSo new_this;
185         new_this.iTu = this->iTu * another.iMau;
186         PhanSo new_another;
187         new_another.iTu = another.iTu * this->iMau;
188         return new_this.iTu >= new_another.iTu;
189     }
```

Hình 12. Nội dung của phương thức operator>=(PhanSo another)

```
bool PhanSo::operator>=(PhanSo another)
{
    PhanSo new_this;
    new_this.iTu = this->iTu * another.iMau;
    PhanSo new_another;
    new_another.iTu = another.iTu * this->iMau;
    return new_this.iTu >= new_another.iTu;
}
```

Bảng 16. Nội dung của phương thức operator>=(PhanSo another)

1.1.11 – Phương thức operator<=(PhanSo another):

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Nội dung: Phép so sánh bé hơn hoặc bằng, nếu phân số thứ nhất bé hơn hoặc bằng phân số thứ 2 thì kết quả được trả về.

```
191 bool PhanSo::operator<=(PhanSo another)
192 {
193     PhanSo new_this;
194     new_this.iTu = this->iTu * another.iMau;
195     PhanSo new_another;
196     new_another.iTu = another.iTu * this->iMau;
197     return new_this.iTu <= new_another.iTu;
198 }
```

Hình 13. Nội dung của phương thức operator<=(PhanSo another)

```
bool PhanSo::operator<=(PhanSo another)
{
    PhanSo new_this;
    new_this.iTu = this->iTu * another.iMau;
    PhanSo new_another;
    new_another.iTu = another.iTu * this->iMau;
    return new_this.iTu <= new_another.iTu;
}
```

Bảng 17. Nội dung của phương thức operator<=(PhanSo another)

1.1.12 – Phương thức operator>(PhanSo another):

- Nội dung: Phép so sánh lớn hơn, nếu phân số thứ nhất lớn hơn phân số thứ 2 thì kết quả được trả về.

```
200 bool PhanSo::operator>(PhanSo another)
201 {
202     PhanSo new_this;
203     new_this.iTu = this->iTu * another.iMau;
204     PhanSo new_another;
205     new_another.iTu = another.iTu * this->iMau;
206     return new_this.iTu > new_another.iTu;
207 }
```

Hình 14. Nội dung của phương thức operator>(PhanSo another)

```
bool PhanSo::operator>(PhanSo another)
```

```

{
    PhanSo new_this;
    new_this.iTu = this->iTu * another.iMau;
    PhanSo new_another;
    new_another.iTu = another.iTu * this->iMau;
    return new_this.iTu > new_another.iTu;
}

```

*Bảng 18. Nội dung của phương thức operator>(PhanSo another)***1.1.13 – Phương thức operator<(PhanSo another):**

- Nội dung: Phép so sánh bé hơn, nếu phân số thứ nhất bé hơn phân số thứ 2 thì kết quả được trả về.

```

209    bool PhanSo::operator<(PhanSo another)
210    {
211        PhanSo new_this;
212        new_this.iTu = this->iTu * another.iMau;
213        PhanSo new_another;
214        new_another.iTu = another.iTu * this->iMau;
215        return new_this.iTu < new_another.iTu;
216    }

```

Hình 15. Nội dung của phương thức operator<(PhanSo another)

```

bool PhanSo::operator<(PhanSo another)
{
    PhanSo new_this;
    new_this.iTu = this->iTu * another.iMau;
    PhanSo new_another;
    new_another.iTu = another.iTu * this->iMau;
    return new_this.iTu < new_another.iTu;
}

```

*Bảng 19. Nội dung của phương thức operator<(PhanSo another)***1.1.14 – Phương thức operator>>(istream &in, PhanSo &ps):**

- Nội dung: Overload toán tử >>

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
47   istream &operator>>(istream &in, PhanSo &ps)
48   {
49       cout << "Nhap vao Tu";
50       in >> ps.iTu;
51       cout << "Nhap vao Mau";
52       in >> ps.iMau;
53       return in;
54   }
```

Hình 16. Nội dung của phương thức `operator>>(istream &in, PhanSo &ps)`

```
istream &operator>>(istream &in, PhanSo &ps)
{
    cout << "Nhap vao Tu";
    in >> ps.iTu;
    cout << "Nhap vao Mau";
    in >> ps.iMau;
    return in;
}
```

Bảng 20. Nội dung của phương thức `operator>>(istream &in, PhanSo &ps)`

1.1.15 – Phương thức `operator<<(ostream &out, PhanSo &ps)`:

- Nội dung: Overload toán tử `<<`

```
56   ostream &operator<<(ostream &out, PhanSo &ps)
57   {
58       out << ps.iTu << "/" << ps.iMau;
59       return out;
60   }
```

Hình 17. Nội dung của phương thức `operator<<(ostream &out, PhanSo &ps)`

```
ostream &operator<<(ostream &out, PhanSo &ps)
{
    out << ps.iTu << "/" << ps.iMau;
    return out;
}
```

Bảng 21. Nội dung của phương thức `operator<<(ostream &out, PhanSo &ps)`

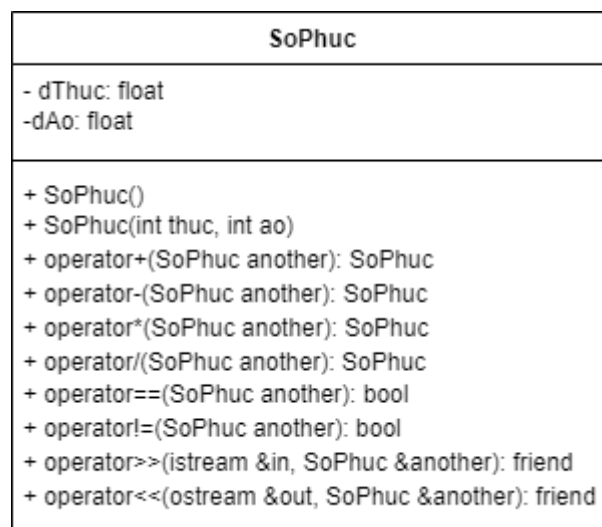
IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Bài tập 2: Xây dựng lớp số phức:

- Thuộc tính: dThuc, dAo
- Phương thức: SoPhuc(), SoPhuc (int thuc, int ao)
- Thực hiện các phương thức operator: +, -, *, /, ==, !=, >>, <<

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm main()

Class diagram của lớp SoPhuc:



Hình 18. Class diagram của lớp SoPhuc

Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
1  #include <iostream>      You, 4 weeks ago • First commit - com
2  #include <iomanip>
3
4  using namespace std;
5
6  You, 4 weeks ago | 1 author (You)
7  class SoPhuc
8  {
9  private:
10     float dThuc, dAo;
11 public:
12     SoPhuc();
13     SoPhuc(int thuc, int ao);
14     SoPhuc operator+(SoPhuc another);
15     SoPhuc operator-(SoPhuc another);
16     SoPhuc operator*(SoPhuc another);
17     SoPhuc operator/(SoPhuc another);
18     bool operator==(SoPhuc another);
19     bool operator!=(SoPhuc another);
20     friend istream &operator>>(istream &in, SoPhuc &another);
21     friend ostream &operator<<(ostream &out, SoPhuc &another);
22 };
```

Hình 19. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc

```
#include <iostream>
#include <iomanip>

using namespace std;

class SoPhuc
{
private:
    float dThuc, dAo;

public:
    SoPhuc();
    SoPhuc(int thuc, int ao);
    SoPhuc operator+(SoPhuc another);
    SoPhuc operator-(SoPhuc another);
    SoPhuc operator*(SoPhuc another);
```

```

SoPhuc operator/(SoPhuc another);
bool operator==(SoPhuc another);
bool operator!=(SoPhuc another);
friend istream &operator>>(istream &in, SoPhuc &another);
friend ostream &operator<<(ostream &out, SoPhuc &another);
};

```

Bảng 22. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc

2.1 – Phương thức của class SoPhuc:

2.1.1 – Phương thức SoPhuc()

- Nội dung: Gán giá trị ban đầu của dThuc, dAo = 0.

```

24  SoPhuc::SoPhuc()
25  {
26  |      dThuc = 0;
27  |      dAo = 0;
28  |  }

```

Hình 20. Nội dung của phương thức SoPhuc()

```

SoPhuc::SoPhuc()
{
    dThuc = 0;
    dAo = 0;
}

```

Bảng 23. Nội dung của phương thức SoPhuc()

2.1.2 – Phương thức SoPhuc(int thuc, int ao)

- Nội dung: Gán giá trị nhập vào biến dThuc, dAo.

```

30  SoPhuc::SoPhuc(int thuc, int ao)
31  {
32  |      dThuc = thuc;
33  |      dAo = ao;
34  |  }

```

Hình 21. Nội dung của phương thức SoPhuc(int thuc, int ao)

```

SoPhuc::SoPhuc(int thuc, int ao)

```

```
{
    dThuc = thuc;
    dAo = ao;
}
```

Bảng 24. Nội dung của phương thức SoPhuc(int thuc, int ao)

2.1.3 – Phương thức operator+(SoPhuc another)

- Input: Số phức muốn cộng.
- Output: Tổng 2 số phức sau khi cộng.
- Hướng giải quyết: Cộng phần ảo, phần thực riêng rẽ => trả về kết quả sau khi cộng.

```
36  SoPhuc SoPhuc::operator+(SoPhuc another)
37  {
38      SoPhuc kq;
39      kq.dThuc = dThuc + another.dThuc;
40      kq.dAo = dAo + another.dAo;
41      cout << "Tong cua 2 so phuc la: " << kq.dThuc << " + " << kq.dAo <<
      "i\n";
42      return kq;
43  }
```

Hình 22. Nội dung của phương thức operator+(SoPhuc another)

```
SoPhuc SoPhuc::operator+(SoPhuc another)
{
    SoPhuc kq;
    kq.dThuc = dThuc + another.dThuc;
    kq.dAo = dAo + another.dAo;
    cout << "Tong cua 2 so phuc la: " << kq.dThuc << " + " << kq.dAo <<
    "i\n";
    return kq;
}
```

Bảng 25. Nội dung của phương thức operator+(SoPhuc another)

Input: 1 2

//Giả sử số phức hiện tại là $1 + 2i$

Output: $2 + 2i$

Bảng 26. Ví dụ Input/Output của phương thức operator+(SoPhuc another)

2.1.4 – Phương thức operator-(SoPhuc another)

- Input: Số phức muốn trừ.
- Output: Hiệu 2 số phức sau khi trừ.
- Hướng giải quyết: Trừ phần ảo, phần thực riêng rẽ => trả về kết quả sau khi trừ.

```

45  SoPhuc SoPhuc::operator-(SoPhuc another)
46  {
47      SoPhuc kq;
48      kq.dThuc = dThuc - another.dThuc;
49      kq.dAo = dAo - another.dAo;
50      cout << "Hiệu của 2 số phức là: " << kq.dThuc << " + " << kq.dAo <<
      "i\n";
51      return kq;
52  }

```

Hình 23. Nội dung của phương thức operator-(SoPhuc another)

```

SoPhuc SoPhuc::operator-(SoPhuc another)
{
    SoPhuc kq;
    kq.dThuc = dThuc - another.dThuc;
    kq.dAo = dAo - another.dAo;
    cout << "Hiệu của 2 số phức là: " << kq.dThuc << " + " << kq.dAo <<
    "i\n";
    return kq;
}

```

Bảng 27. Nội dung của phương thức operator-(SoPhuc another)

Input: 1 2

//Giả sử số phức hiện tại là $1 + 2i$ Output: $0 + 0i$ *Bảng 28. Ví dụ về Input/Output của phương thức operator-(SoPhuc another)***2.1.5 – Phương thức operator*(SoPhuc another)**

- Input: Số phức muốn nhân.
- Output: Tích số phức.
- Hướng giải quyết: áp dụng công thức nhân số phức

$$z_1.z_2 = (ac - bd) + (ad + bc)i$$

```

54  SoPhuc SoPhuc::operator*(SoPhuc another)
55  {
56      SoPhuc kq;
57      kq.dThuc = dThuc * another.dThuc - dAo * another.dAo;
58      kq.dAo = dThuc * another.dAo + another.dThuc * dAo;
59      cout << "Tích của 2 số phức là: " << kq.dThuc << " + " << kq.dAo << 'i'
        << endl;
60      return kq;
61  }

```

Hình 24. Nội dung của phương thức operator*(SoPhuc another)

```

SoPhuc SoPhuc::operator*(SoPhuc another)
{
    SoPhuc kq;
    kq.dThuc = dThuc * another.dThuc - dAo * another.dAo;
    kq.dAo = dThuc * another.dAo + another.dThuc * dAo;
    cout << "Tích của 2 số phức là: " << kq.dThuc << " + " << kq.dAo << 'i'
    << endl;
    return kq;
}

```

Bảng 29. Nội dung của phương thức operator*(SoPhuc another)

Input: 2 3

//Giả sử số phức hiện tại là $1 + 2i$

Output: $-4 + 7i$

Bảng 30. Ví dụ Input/Output của phương thức operator*(SoPhuc another)

2.1.6 – Phương thức operator/(SoPhuc another)

- Input: Số phức muốn chia .
- Output: Thương số phức.
- Hướng giải quyết: áp dụng công thức chia số phức

$$\frac{z_1}{z_2} = \frac{z_1 \cdot \overline{z_2}}{|z_2|^2} = \frac{ac + bd}{c^2 + d^2} + \frac{bc - ad}{c^2 + d^2} \cdot i \quad (\text{với } z_2 \neq 0)$$

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
63 SoPhuc SoPhuc::operator/(SoPhuc another)
64 {
65     SoPhuc kq;
66     kq.dThuc = (dThuc * another.dThuc + dAo * another.dAo) / (another.dThuc
        * another.dThuc + another.dAo * another.dAo);
67     kq.dAo = (dAo * another.dThuc - dThuc * another.dAo) / (another.dThuc *
        another.dThuc + another.dAo * another.dAo);
68     cout << "Thuong cua 2 so phuc la: " << fixed << setprecision(2) << kq.
        dThuc << " + " << kq.dAo << 'i' << endl;
69     return kq;
70 }
```

Hình 25. Nội dung của phương thức operator/(SoPhuc another)

```
SoPhuc SoPhuc::operator/(SoPhuc another)
{
    SoPhuc kq;
    kq.dThuc = (dThuc * another.dThuc + dAo * another.dAo) / (another.dThuc
    * another.dThuc + another.dAo * another.dAo);
    kq.dAo = (dAo * another.dThuc - dThuc * another.dAo) / (another.dThuc *
    another.dThuc + another.dAo * another.dAo);
    cout << "Thuong cua 2 so phuc la: " << fixed << setprecision(2) <<
    kq.dThuc << " + " << kq.dAo << 'i' << endl;
    return kq;
}
```

Bảng 31. Nội dung của phương thức operator/(SoPhuc another)

Input: 2 3

//Giả sử số phức hiện tại là $1 + 2i$

Output: $0.62 + 0.08i$

Bảng 32. Ví dụ Input/Output của phương thức operator/(SoPhuc another)

2.1.7 – Phương thức operator==(SoPhuc another)

- Nội dung: So sánh bằng 2 số phức, nếu 2 số phức bằng nhau thì trả về kết quả true.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
72  ✓ bool SoPhuc::operator==(SoPhuc another)
73      {
74      |     return dThuc == another.dThuc && dAo == another.dAo;
75      }
```

Hình 26. Nội dung của phương thức operator==(SoPhuc another)

```
bool SoPhuc::operator==(SoPhuc another)
{
    return dThuc == another.dThuc && dAo == another.dAo;
}
```

Bảng 33. Nội dung của phương thức operator==(SoPhuc another)

2.1.8 – Phương thức operator!=(SoPhuc another)

- Nội dung: So sánh không bằng, nếu 2 số phức không bằng nhau thì trả về true.

```
77  ✓ bool SoPhuc::operator!=(SoPhuc another)
78      {
79      |     return !(*this == another);
80      }
```

Hình 27. Nội dung của phương thức operator!=(SoPhuc another)

```
bool SoPhuc::operator!=(SoPhuc another)
{
    return !(*this == another);
}
```

Bảng 34. Nội dung của phương thức operator!=(SoPhuc another)

2.1.9 – Phương thức operator>>(istream &in, SoPhuc &another)

- Nội dung: Overload toán tử >>

```
82  ✓ istream &operator>>(istream &in, SoPhuc &another)
83      {
84      |     cout << "Nhập vào phần Thực: ";
85      |     in >> another.dThuc;
86      |     cout << "Nhập vào phần Ao: ";
87      |     in >> another.dAo;
88      |     return in;
89      }
```

Hình 28. Nội dung phương thức operator>>(istream &in, SoPhuc &another)

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
istream &operator>>(istream &in, SoPhuc &another)
{
    cout << "Nhap vao phan Thuc: ";
    in >> another.dThuc;
    cout << "Nhap vao phan Ao: ";
    in >> another.dAo;
    return in;
}
```

Bảng 35. Nội dung phương thức operator>>(istream &in, SoPhuc &another)

2.1.10 – Phương thức operator<<(ostream &out, SoPhuc &another)

- Nội dung: Overload toán tử <<

```
91  ostream &operator<<(ostream &out, SoPhuc &another)
92  {
93      out << another.dThuc << " + " << another.dAo << "i\n";
94      return out;
95  }
```

Hình 29. Nội dung của phương thức operator<<(ostream &out, SoPhuc &another)

```
ostream &operator<<(ostream &out, SoPhuc &another)
{
    out << another.dThuc << " + " << another.dAo << "i\n";
    return out;
}
```

Bảng 36. Nội dung của phương thức operator<<(ostream &out, SoPhuc &another)

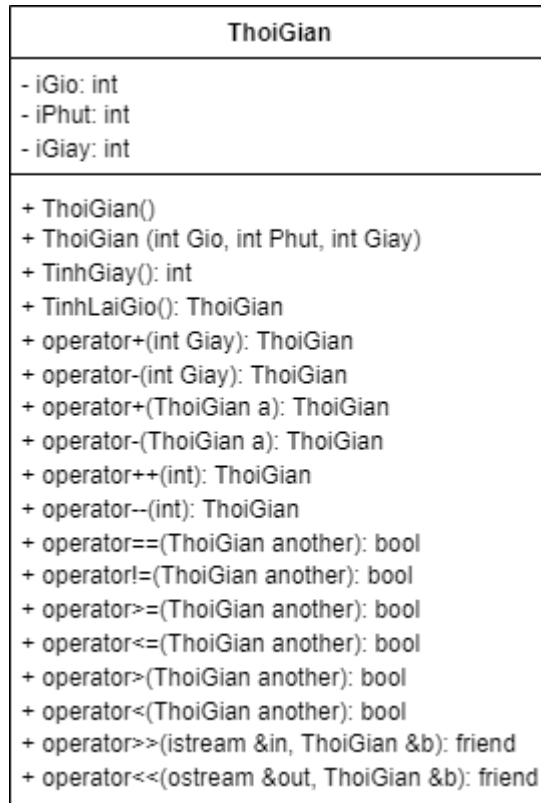
Bài tập 3. Xây dựng lớp thời gian

- Thuộc tính: iGio, iPhut, iGiay
- Phương thức: ThoiGian(), ThoiGian (int Gio, int Phut, int Giay), TinhGiay(), TinhLaiGio(int Giay)
- Thực hiện các phương thức operator: +(int Giay), -(int Giay), +(ThoiGian a), (ThoiGian a), ++, --, ==, !=, >=, <=, >, <, <<

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm main()

Class diagram của lớp ThoiGian:



Hình 30. Class diagram của lớp ThoiGian

Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
1  #include <iostream>
2
3  using namespace std;
4
   You, 16 minutes ago | 1 author (You)
5  class ThoiGian
6  {
7  private:
8      int iGio, iPhut, iGiay;
9
10 public:
11     ThoiGian();
12     ThoiGian(int Gio, int Phut, int Giay);
13     int TinhGiay();
14     ThoiGian TinhLaiGio();
15     ThoiGian operator+(int Giay);
16     ThoiGian operator-(int Giay);
17     ThoiGian operator+(ThoiGian a);
18     ThoiGian operator-(ThoiGian a);
19     ThoiGian operator++(int);
20     ThoiGian operator--(int);
21     bool operator==(ThoiGian another);
22     bool operator!=(ThoiGian another);
23     bool operator>=(ThoiGian another);
24     bool operator<=(ThoiGian another);
25     bool operator>(ThoiGian another);
26     bool operator<(ThoiGian another);
27     friend istream &operator>>(istream &in, ThoiGian &b);
28     friend ostream &operator<<(ostream &out, ThoiGian &b);
29 };

```

Hình 31. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian

```
#include <iostream>

using namespace std;

class ThoiGian
{

```

```

private:
    int iGio, iPhut, iGiay;

public:
    ThoiGian();
    ThoiGian(int Gio, int Phut, int Giay);
    int TinhGiay();
    ThoiGian TinhLaiGio();
    ThoiGian operator+(int Giay);
    ThoiGian operator-(int Giay);
    ThoiGian operator+(ThoiGian a);
    ThoiGian operator-(ThoiGian a);
    ThoiGian operator++(int);
    ThoiGian operator--(int);
    bool operator==(ThoiGian another);
    bool operator!=(ThoiGian another);
    bool operator>=(ThoiGian another);
    bool operator<=(ThoiGian another);
    bool operator>(ThoiGian another);
    bool operator<(ThoiGian another);
    friend istream &operator>>(istream &in, ThoiGian &b);
    friend ostream &operator<<(ostream &out, ThoiGian &b);
};

```

Bảng 37. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian

3.1 – Phương thức của class ThoiGian:

3.1.1 – Phương thức ThoiGian():

- Nội dung: Gán giá trị ban đầu của giờ, phút, giây = 0.

```

48  ✓ ThoiGian::ThoiGian()
49  {
50      iGio = 0;
51      iPhut = 0;
52      iGiay = 0;
53  }

```

Hình 32. Nội dung của phương thức ThoiGian()


```

ThoiGian::ThoiGian()
{
    iGio = 0;
    iPhut = 0;
    iGiay = 0;
}

```

*Bảng 38. Nội dung của phương thức ThoiGian()***3.1.2 – Phương thức ThoiGian(int Gio, int Phut, int Giay):**

- Nội dung: Gán giá trị nhập vào cho các biến giờ, phút, giây.

```

55  ThoiGian::ThoiGian(int Gio, int Phut, int Giay)
56  {
57      iGio = Gio;
58      iPhut = Phut;
59      iGiay = Giay;
60  }

```

Hình 33. Nội dung của phương thức ThoiGian(int Gio, int Phut, int Giay)

```

ThoiGian::ThoiGian(int Gio, int Phut, int Giay)
{
    iGio = Gio;
    iPhut = Phut;
    iGiay = Giay;
}

```

*Bảng 39. Nội dung của phương thức ThoiGian(int Gio, int Phut, int Giay)***3.1.3 – Phương thức TinhGiay():**

- Input: Giờ, phút giây
- Output: Giây
- Hướng giải quyết: Tổng của $3600 * iGio + 60 * iPhut + iGiay$

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
62  int ThoiGian::TinhGiay()  
63  {  
64      return (iGio * 3600 + iPhut * 60 + iGiay);  
65  }
```

Hình 34. Nội dung của phương thức TinhGiay()

```
int ThoiGian::TinhGiay()  
{  
    return (iGio * 3600 + iPhut * 60 + iGiay);  
}
```

Bảng 40. Nội dung của phương thức TinhGiay()

| |
|-----------------|
| Input: 22 59 59 |
| Output: 82799 |

Bảng 41. Ví dụ Input/Output của phương thức TinhGiay()

3.1.4 – Phương thức TinhLaiGio():

Input: Giây.

Output: Thời gian sau khi chuyển từ giây.

Hướng giải quyết: Áp dụng phương thức TinhGiay() sau đó chuyển lại giờ, phút, giây.

```
67  ThoiGian ThoiGian::TinhLaiGio()  
68  {  
69      int temp_Giay = this->TinhGiay();  
70      ThoiGian temp;  
71      temp.iGio = temp_Giay / 3600;  
72      temp_Giay %= 3600;  
73      temp.iPhut = temp_Giay / 60;  
74      temp.iGiay = temp_Giay % 60;  
75      return temp;  
76  }
```

Hình 35. Nội dung của phương thức TinhLaiGio().

```

ThoiGian ThoiGian::TinhLaiGio()
{
    int temp_Giay = this->TinhGiay();
    ThoiGian temp;
    temp.iGio = temp_Giay / 3600;
    temp_Giay %= 3600;
    temp.iPhut = temp_Giay / 60;
    temp.iGiay = temp_Giay % 60;
    return temp;
}

```

Bảng 42. Nội dung của phương thức TinhLaiGio().

| |
|------------------|
| Input: 82799 |
| Output: 22:59:59 |

*Bảng 43. Ví dụ Input/Output của phương thức TinhLaiGio().***3.1.5 – Phương thức operator+(int Giay):**

- Input: n giây muốn cộng
- Output: Thời gian sau khi cộng thêm n giây
- Hướng giải quyết:
 1. Lấy số giây hiện tại cộng thêm n giây.
 2. Chạy vòng lặp **while**, nếu iGiay hiện tại lớn hơn 59 => iGiay trừ cho 60.
 - 2.1. Sau đó tạo điều kiện **if (iPhut < 59)** => iPhut hiện tại ++.
 - 2.2. Điều kiện **else if (iPhut == 59)** thì iPhut = 0.
 - 2.2.1. Tạo điều kiện **if (iGio == 23)** => iGio = 0.
 - 2.2.2. Tạo điều kiện **else if (iGio < 23)** => iGio ++.
 3. Return kết quả

```
222 ThoiGian ThoiGian::operator+(int Giay)
223 {
224     ThoiGian temp = *this;
225     temp.iGiay += Giay;
226     while (temp.iGiay > 59)
227     {
228         temp.iGiay -= 60;
229         if (temp.iPhut < 59)
230         {
231             temp.iPhut++;
232         }
233         else if (temp.iPhut == 59)
234         {
235             temp.iPhut = 0;
236             if (temp.iGio == 23)
237             {
238                 temp.iGio = 0;
239             }
240             else if (temp.iGio < 23)
241             {
242                 temp.iGio++;
243             }
244         }
245     }
246     return temp;
247 }
```

Hình 36. Nội dung của phương thức operator+(int Giay)

```
ThoiGian ThoiGian::operator+(int Giay)
{
    ThoiGian temp = *this;
    temp.iGiay += Giay;
    while (temp.iGiay > 59)
    {
        temp.iGiay -= 60;
        if (temp.iPhut < 59)
        {
            temp.iPhut++;
        }
    }
}
```

```

    }
    else if (temp.iPhut == 59)
    {
        temp.iPhut = 0;
        if (temp.iGio == 23)
        {
            temp.iGio = 0;
        }
        else if (temp.iGio < 23)
        {
            temp.iGio++;
        }
    }
}
return temp;
}

```

Bảng 44. Nội dung của phương thức operator+(int Giay)

| |
|---|
| Input: 1000 |
| //Giả sử thời gian hiện tại là 22:59:59 |
| Output: 23:16:39 |

Bảng 45. Ví dụ Input/Output của phương thức operator+(int Giay)

3.1.6 – Phương thức operator-(int Giay):

- Input: n giây muốn trừ.
- Output: Thời gian sau khi trừ đi n giây
- Hướng giải quyết:
 1. Lấy số giây hiện tại trừ đi n giây.
 2. Chạy vòng lặp **while**, nếu iGiay hiện tại bé hơn 0 => iGiay cộng thêm 60.
 - 2.1. Sau đó tạo điều kiện **if (iPhut > 0 && iPhut <= 59)** => iPhut hiện tại --.
 - 2.2. Điều kiện **else if (iPhut == 0)** thì iPhut = 59.
 - 2.2.1. Tạo điều kiện **if (iGio == 0)** => iGio = 23.
 - 2.2.2. Tạo điều kiện **else if (iGio > 0 && iGio <= 23)** => iGio --.
 3. Return kết quả

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
249 ThoiGian ThoiGian::operator-(int Giay)
250 {
251     ThoiGian temp = *this;
252     temp.iGiay -= Giay;
253     while (temp.iGiay < 0)
254     {
255         temp.iGiay += 60;
256         if (temp.iPhut > 0 && temp.iPhut <= 59)
257         {
258             temp.iPhut--;
259         }
260         else if (temp.iPhut == 0)
261         {
262             temp.iPhut = 59;
263             if (temp.iGio > 0 && temp.iGio <= 23)
264             {
265                 temp.iGio--;
266             }
267             else if (temp.iGio == 0)
268             {
269                 temp.iGio = 23;
270             }
271         }
272     }
273     return temp;
274 }
```

You, 4 weeks ago • First commit - committing

Hình 37. Nội dung của phương thức operator-(int Giay)

```
ThoiGian ThoiGian::operator-(int Giay)
{
    ThoiGian temp = *this;
    temp.iGiay -= Giay;
    while (temp.iGiay < 0)
    {
        temp.iGiay += 60;
        if (temp.iPhut > 0 && temp.iPhut <= 59)
        {
            temp.iPhut--;
        }
    }
}
```

```

    }
    else if (temp.iPhut == 0)
    {
        temp.iPhut = 59;
        if (temp.iGio > 0 && temp.iGio <= 23)
        {
            temp.iGio--;
        }
        else if (temp.iGio == 0)
        {
            temp.iGio = 23;
        }
    }
}
return temp;
}

```

Bảng 46. Nội dung của phương thức operator-(int Giay)

| |
|---|
| Input: 1000 |
| //Giả sử thời gian hiện tại là 22:59:59 |
| Output: 22:43:19 |

*Bảng 47. Ví dụ Input/Output của phương thức operator-(int Giay)***3.1.7 – Phương thức operator+(ThoiGian a):**

- Input: Thời gian a muốn cộng thêm.
- Output: Thời gian sau khi cộng thêm thời gian a
- Hướng giải quyết:
 1. Chuyển đổi thời gian a sang giây.
 2. Áp dụng như **function operator+(int Giay)**.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
166 ThoiGian ThoiGian::operator+(ThoiGian a)
167 {
168     ThoiGian kq = *this;
169     int temp_giay_a = a.iGio * 3600 + a.iPhut * 60 + a.iGiay;
170     kq.iGiay += temp_giay_a;
171     while (kq.iGiay > 59)
172     {
173         kq.iGiay -= 60;
174         if (kq.iPhut < 59)
175         {
176             kq.iPhut++;
177         }
178         else if (kq.iPhut == 59)
179         {
180             kq.iPhut = 0;
181             if (kq.iGio == 23)
182             {
183                 kq.iGio = 0;
184             }
185             else if (kq.iGio < 23)
186             {
187                 kq.iGio++;
188             }
189         }
190     }
191     return kq;
```

Hình 38. Nội dung của phương thức `operator+(ThoiGian a)`

```
ThoiGian ThoiGian::operator+(ThoiGian a)
{
    ThoiGian kq = *this;
    int temp_giay_a = a.iGio * 3600 + a.iPhut * 60 + a.iGiay;
    kq.iGiay += temp_giay_a;
    while (kq.iGiay > 59)
    {
        kq.iGiay -= 60;
        if (kq.iPhut < 59)
        {
            kq.iPhut++;
        }
        else if (kq.iPhut == 59)
        {
            kq.iPhut = 0;
```



```

        if (kq.iGio == 23)
        {
            kq.iGio = 0;
        }
        else if (kq.iGio < 23)
        {
            kq.iGio++;
        }
    }
}
return kq;
}

```

Bảng 48. Nội dung của phương thức operator+(ThoiGian a)

| |
|---|
| Input: 21:59:59 |
| //Giả sử thời gian hiện tại là 22:59:59 |
| Output: 20:59:58 |

*Bảng 49. Ví dụ Input/Output của phương thức operator-(ThoiGian a)***3.1.8 – Phương thức operator-(ThoiGian a):**

- Input: Thời gian a muốn trừ đi.
- Output: Thời gian hiện tại sau khi trừ đi thời gian a.
- Hướng giải quyết:
 1. Chuyển đổi thời gian a sang giây.
 2. Áp dụng như **function operator-(int Giay)**.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
194 ThoiGian ThoiGian::operator-(ThoiGian a)
195 {
196     ThoiGian kq = *this;
197     int temp_giay_a = a.iGio * 3600 + a.iPhut * 60 + a.iGiay;
198     kq.iGiay -= temp_giay_a;
199     while (kq.iGiay < 0)
200     {
201         kq.iGiay += 60;
202         if (kq.iPhut > 0 && kq.iPhut <= 59)
203         {
204             kq.iPhut--;
205         }
206         else if (kq.iPhut == 0)
207         {
208             kq.iPhut = 59;
209             if (kq.iGio > 0 && kq.iGio <= 23)
210             {
211                 kq.iGio--;
212             }
213             else if (kq.iGio == 0)
214             {
215                 kq.iGio = 23;
216             }
217         }
218     }
219     return kq;
```

Hình 39. Nội dung của phương thức operator-(ThoiGian a)

```
ThoiGian ThoiGian::operator-(ThoiGian a)
{
    ThoiGian kq = *this;
    int temp_giay_a = a.iGio * 3600 + a.iPhut * 60 + a.iGiay;
    kq.iGiay -= temp_giay_a;
    while (kq.iGiay < 0)
    {
        kq.iGiay += 60;
        if (kq.iPhut > 0 && kq.iPhut <= 59)
        {
            kq.iPhut--;
        }
        else if (kq.iPhut == 0)
        {
```

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
kq.iPhut = 59;
if (kq.iGio > 0 && kq.iGio <= 23)
{
    kq.iGio--;
}
else if (kq.iGio == 0)
{
    kq.iGio = 23;
}
}
return kq;
}
```

Bảng 50. Nội dung của phương thức operator-(ThoiGian a)

Input: 21:59:59

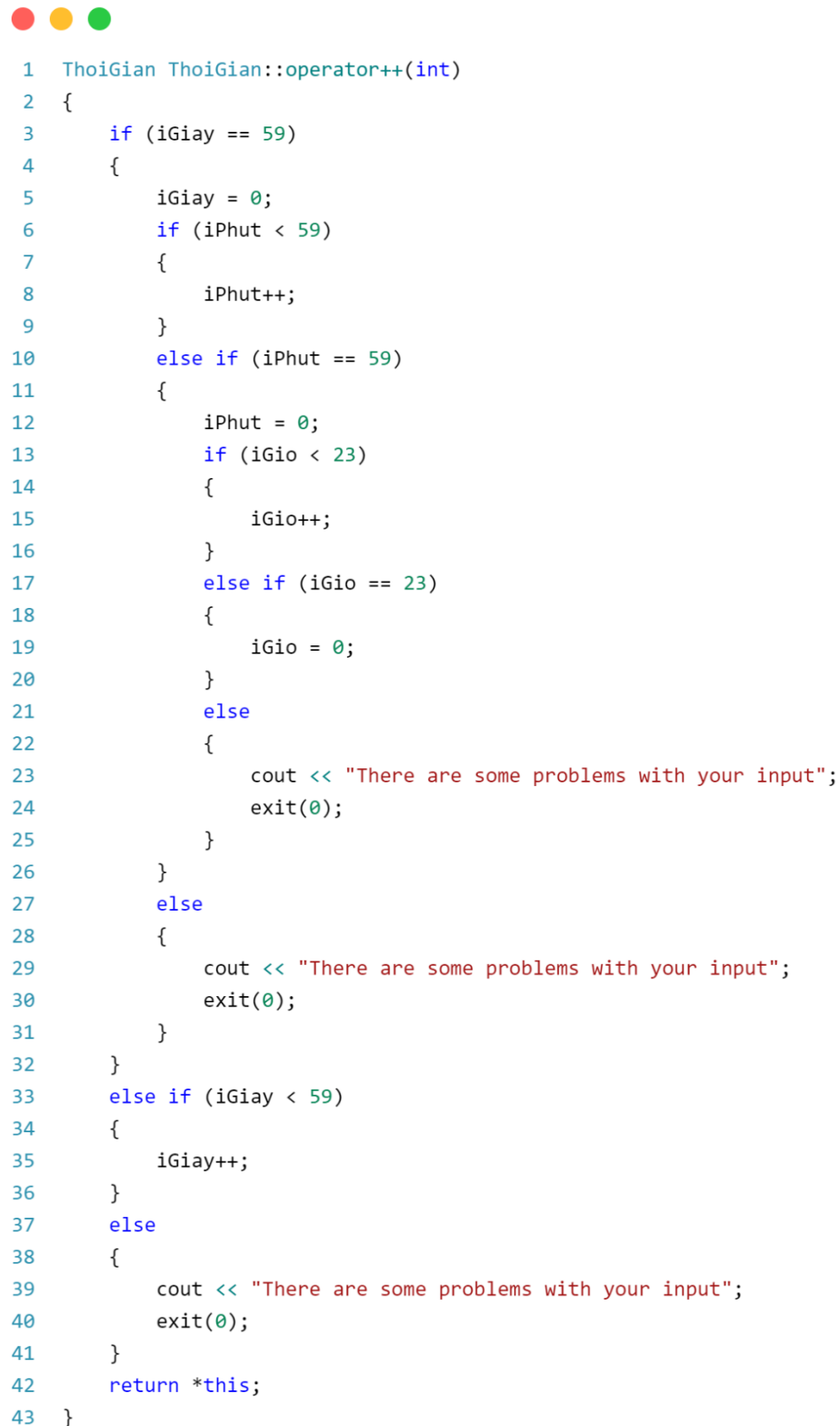
//Giả sử thời gian hiện tại là 22:59:59

Output: 1:0:0

Bảng 51. Ví dụ Input/Output của phương thức operator-(ThoiGian a)

3.1.9 – Phương thức operator++(int):

- Nội dung: Thời gian sau khi cộng thêm một giây
- Hướng giải quyết:
 1. Tạo điều kiện **if (iGiay < 59)** => iGiay hiện tại ++.
 2. Điều kiện **else if (iGiay == 59)** thì iGiay = 0.
 - 2.1. Tạo điều kiện **if (iPhut < 59)** => iPhut++.
 - 2.2. Tạo điều kiện **else if (iPhut == 59)** => iPhut = 0.
 - 2.2.1. Tạo điều kiện **if (iGio < 23)** => iGio ++.
 - 2.2.1. Tạo điều kiện **else if (iGio == 23)** => iGio = 0.



```

1  ThoiGian ThoiGian::operator++(int)
2  {
3      if (iGiay == 59)
4      {
5          iGiay = 0;
6          if (iPhut < 59)
7          {
8              iPhut++;
9          }
10         else if (iPhut == 59)
11         {
12             iPhut = 0;
13             if (iGio < 23)
14             {
15                 iGio++;
16             }
17             else if (iGio == 23)
18             {
19                 iGio = 0;
20             }
21             else
22             {
23                 cout << "There are some problems with your input";
24                 exit(0);
25             }
26         }
27         else
28         {
29             cout << "There are some problems with your input";
30             exit(0);
31         }
32     }
33     else if (iGiay < 59)
34     {
35         iGiay++;
36     }
37     else
38     {
39         cout << "There are some problems with your input";
40         exit(0);
41     }
42     return *this;
43 }

```

Hình 40. Nội dung phương thức operator++(int)

```
ThoiGian ThoiGian::operator++(int)
{
    if (iGiay == 59)
    {
        iGiay = 0;
        if (iPhut < 59)
        {
            iPhut++;
        }
        else if (iPhut == 59)
        {
            iPhut = 0;
            if (iGio < 23)
            {
                iGio++;
            }
            else if (iGio == 23)
            {
                iGio = 0;
            }
            else
            {
                cout << "There are some problems with your input";
                exit(0);
            }
        }
        else
        {
            cout << "There are some problems with your input";
            exit(0);
        }
    }
    else if (iGiay < 59)
    {
        iGiay++;
    }
    else
    {
        cout << "There are some problems with your input";
        exit(0);
    }
    return *this;
}
```

Bảng 52. Nội dung phương thức operator++(int)

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

| |
|-----------------|
| Input: 22:59:59 |
| Output: 23:0:0 |

Bảng 53. Ví dụ Input/Output của phương thức `operator++(int)`

3.1.10 – Phương thức `operator--(int)`:

- Nội dung: Thời gian sau khi trừ đi một giây
- Hướng giải quyết:
 1. Tạo điều kiện **if (iGiay > 0 && iGiay <= 59)** => iGiay hiện tại --.
 2. Điều kiện **else if (iGiay == 0)** thì iGiay = 59.
 - 2.1. Tạo điều kiện **if (iPhut > 0 && iPhut <= 23)** => iPhut--.
 - 2.2. Tạo điều kiện **else if (iPhut == 0)** => iPhut = 59.
 - 2.2.1. Tạo điều kiện **if (iGio > 0 && iGio <= 23)** => iGio --.
 - 2.2.1. Tạo điều kiện **else if (iGio == 0)** => iGio = 23.

```
1  ThoiGian ThoiGian::operator--(int)
2  {
3      if (iGiay == 0)
4      {
5          iGiay = 59;
6          if (iPhut > 0 && iPhut <= 59)
7          {
8              iPhut--;
9          }
10         else if (iPhut == 0)
11         {
12             iPhut = 59;
13             if (iGio > 0 && iGio <= 23)
14             {
15                 iGio--;
16             }
17             else if (iGio == 0)
18             {
19                 iGio = 23;
20             }
21             else
22             {
23                 cout << "There are some problems with your input";
24                 exit(0);
25             }
26         }
27         else
28         {
29             cout << "There are some problems with your input";
30             exit(0);
31         }
32     }
33     else if (iGiay > 0 && iGiay <= 59)
34     {
35         iGiay--;
36     }
37     else
38     {
39         cout << "There are some problems with your input";
40         exit(0);
41     }
42     return *this;
43 }
```

Hình 41. Nội dung của phương thức operator--(int)

```
ThoiGian ThoiGian::operator--(int)
{
    if (iGiay == 0)
```

```
{
    iGiay = 59;
    if (iPhut > 0 && iPhut <= 59)
    {
        iPhut--;
    }
    else if (iPhut == 0)
    {
        iPhut = 59;
        if (iGio > 0 && iGio <= 23)
        {
            iGio--;
        }
        else if (iGio == 0)
        {
            iGio = 23;
        }
        else
        {
            cout << "There are some problems with your input";
            exit(0);
        }
    }
    else
    {
        cout << "There are some problems with your input";
        exit(0);
    }
}
else if (iGiay > 0 && iGiay <= 59)
{
    iGiay--;
}
else
{
    cout << "There are some problems with your input";
    exit(0);
}
return *this;
}
```

Bảng 54. Nội dung của phương thức operator--(int)

Input: 22:59:59

Bảng 55. Ví dụ Input/Output của phương thức operator--(int)

3.1.11 – Phương thức operator==(ThoiGian another):

- Nội dung: So sánh bằng, nếu thời gian hiện tại bằng thời gian **another** thì return **true**.

```

276  bool ThoiGian::operator==(ThoiGian another)
277  {
278      return iGio == another.iGio && iPhut == another.iPhut && iGiay ==
279      another.iGiay;
  
```

Hình 42. Nội dung của phương thức operator==(ThoiGian another)

```

bool ThoiGian::operator==(ThoiGian another)
{
    return iGio == another.iGio && iPhut == another.iPhut && iGiay ==
another.iGiay;
}
  
```

Bảng 56. Nội dung của phương thức operator==(ThoiGian another)

3.1.12 - Phương thức operator!=(ThoiGian another):

- Nội dung: So sánh không bằng, nếu thời gian hiện tại không bằng thời gian **another** thì return **true**.

```

281  bool ThoiGian::operator!=(ThoiGian another)
282  {
283      return !(*this == another);
284  }
  
```

Hình 43. Nội dung của phương thức operator!=(ThoiGian another)

```

bool ThoiGian::operator!=(ThoiGian another)
{
    return !(*this == another);
}
  
```

Bảng 57. Nội dung của phương thức operator!=(ThoiGian another)

3.1.13 - Phương thức operator>=(ThoiGian another):

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Nội dung: So sánh lớn hơn bằng, nếu thời gian hiện tại lớn hơn hoặc bằng thời gian **another** thì return **true**.
- Hướng giải quyết: chuyển thời gian hiện tại và **another** sang giây sau đó so sánh.

```
286 bool ThoiGian::operator>=(ThoiGian another)
287 {
288     return this->TinhGiay() >= another.TinhGiay();
289 }
```

Hình 44. Nội dung của phương thức operator>=(ThoiGian another)

```
bool ThoiGian::operator>=(ThoiGian another)
{
    return this->TinhGiay() >= another.TinhGiay();
}
```

Bảng 58. Nội dung của phương thức operator>=(ThoiGian another)

3.1.14 - Phương thức operator<=(ThoiGian another):

- Nội dung: So sánh bé hơn hoặc bằng, nếu thời gian hiện tại bé hơn hoặc bằng thời gian **another** thì return **true**.
- Hướng giải quyết: Chuyển thời gian hiện tại và thời gian **another** sang giây sau đó so sánh.

```
291 bool ThoiGian::operator<=(ThoiGian another)
292 {
293     return this->TinhGiay() <= another.TinhGiay();
294 }
```

Hình 45. Nội dung của phương thức operator<=(ThoiGian another)

```
bool ThoiGian::operator<=(ThoiGian another)
{
    return this->TinhGiay() <= another.TinhGiay();
}
```

Bảng 59. Nội dung của phương thức operator<=(ThoiGian another)

3.1.15 - Phương thức operator>(ThoiGian another):

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Nội dung: So sánh lớn hơn, nếu thời gian hiện tại lớn hơn thời gian **another** thì return **true**.
- Hướng giải quyết: chuyển thời gian hiện tại và **another** sang giây sau đó so sánh.

```
301 bool ThoiGian::operator>(ThoiGian another)
302 {
303     return this->TinhGiay() > another.TinhGiay();
304 }
```

Hình 46. Nội dung của phương thức operator>(ThoiGian another)

```
bool ThoiGian::operator>(ThoiGian another)
{
    return this->TinhGiay() > another.TinhGiay();
}
```

Bảng 60. Nội dung của phương thức operator>(ThoiGian another)

3.1.16 - Phương thức operator<(ThoiGian another):

- Nội dung: So sánh bé hơn, nếu thời gian hiện tại bé hơn thời gian **another** thì return **true**.
- Hướng giải quyết: chuyển thời gian hiện tại và **another** sang giây sau đó so sánh.

```
296 bool ThoiGian::operator<(ThoiGian another)
297 {
298     return this->TinhGiay() < another.TinhGiay();
299 }
```

Hình 47. Nội dung của phương thức operator<(ThoiGian another)

```
bool ThoiGian::operator<(ThoiGian another)
{
    return this->TinhGiay() < another.TinhGiay();
}
```

Bảng 61. Nội dung của phương thức operator<(ThoiGian another)

3.1.17 - Phương thức operator>>(istream &in, ThoiGian &b):

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Nội dung: Overload toán tử nhập >>

```
31  istream &operator>>(istream &in, ThoiGian &b)
32  {
33      cout << "Nhap gio: ";
34      in >> b.iGio;
35      cout << "Nhap phut: ";
36      in >> b.iPhut;
37      cout << "Nhap giay: ";
38      in >> b.iGiay;
39      return in;
40  }
```

Hình 48. Nội dung của phương thức operator>>(istream &in, ThoiGian &b)

```
istream &operator>>(istream &in, ThoiGian &b)
{
    cout << "Nhap gio: ";
    in >> b.iGio;
    cout << "Nhap phut: ";
    in >> b.iPhut;
    cout << "Nhap giay: ";
    in >> b.iGiay;
    return in;
}
```

Bảng 62. Nội dung của phương thức operator>>(istream &in, ThoiGian &b)

3.1.18 - Phương thức operator<<(ostream &out, ThoiGian &b):

- Nội dung: Overload toán tử xuất

```
42  ostream &operator<<(ostream &out, ThoiGian &b)
43  {
44      out << b.iGio << ":" << b.iPhut << ":" << b.iGiay;
45      return out;
46  }
```

Hình 49. Nội dung của phương thức operator<<(ostream &out, ThoiGian &b)

```
ostream &operator<<(ostream &out, ThoiGian &b)
{
    out << b.iGio << ":" << b.iPhut << ":" << b.iGiay;
}
```

```

    return out;
}

```

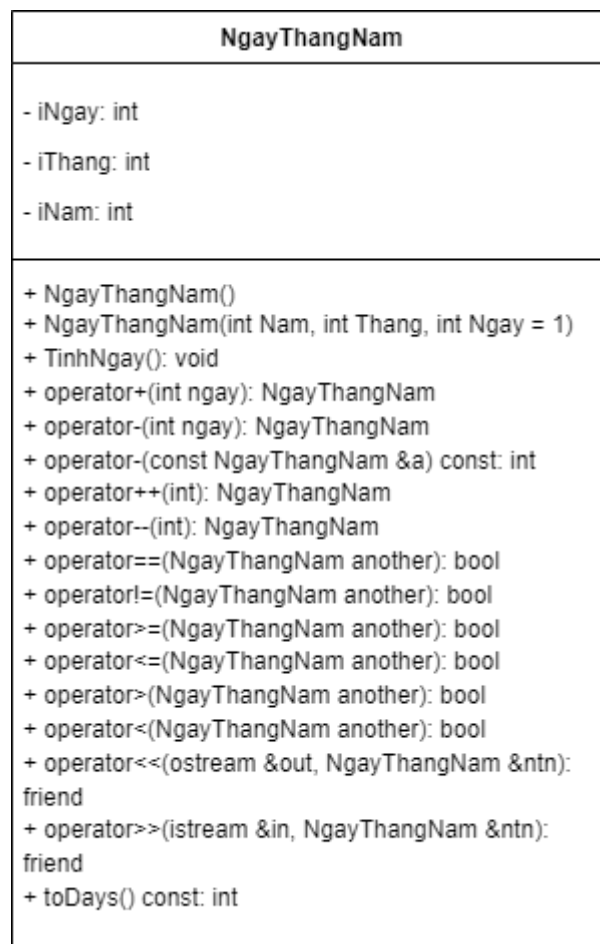
Bảng 63. Nội dung của phương thức `operator<<(ostream &out, ThoiGian &b)`

Bài tập 4. Xây dựng lớp ngày tháng năm

- Thuộc tính: `iNgay`, `iThang`, `iNam`
- Phương thức: `NgayThangNam()`, `NgayThangNam (int Nam, int Thang, int Ngay = 1)`, `TinhNgay()`
- Thực hiện các phương thức operator: `+(int ngay)`, `-(int ngay)`, `-(NgayThangNam a)`, `++`, `--`, `==`, `!=`, `>=`, `<=`, `>`, `<`, `<<`

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm `main()`

Class diagram của lớp `NgayThangNam`:

Hình 50. Class diagram của lớp `NgayThangNam`

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam.

```
1  #include <iostream>
2  #include <algorithm>
3  using namespace std;
4
5  You, 4 weeks ago | 1 author (You)
6  class NgayThangNam
7  {
8  private:
9      int iNgay, iThang, iNam;
10 public:
11     NgayThangNam();
12     NgayThangNam(int Nam, int Thang, int Ngay = 1);
13     void TinhNgay();
14     NgayThangNam operator+(int ngay);
15     NgayThangNam operator-(int ngay);
16     int operator-(const NgayThangNam &a) const;
17     NgayThangNam operator++(int);
18     NgayThangNam operator--(int);
19     bool operator==(NgayThangNam another);
20     bool operator!=(NgayThangNam another);
21     bool operator>=(NgayThangNam another);
22     bool operator<=(NgayThangNam another);
23     bool operator>(NgayThangNam another);
24     bool operator<(NgayThangNam another);
25     friend ostream &operator<<(ostream &out, NgayThangNam &ntn);
26     friend istream &operator>>(istream &in, NgayThangNam &ntn);
27     int toDays() const;
28 };
```

Hình 51. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam

```
#include <iostream>
#include <algorithm>
using namespace std;

class NgayThangNam
{
private:
    int iNgay, iThang, iNam;

public:
```

```

    NgayThangNam();
    NgayThangNam(int Nam, int Thang, int Ngay = 1);
    void TinhNgay();
    NgayThangNam operator+(int ngay);
    NgayThangNam operator-(int ngay);
    int operator-(const NgayThangNam &a) const;
    NgayThangNam operator++(int);
    NgayThangNam operator--(int);
    bool operator==(NgayThangNam another);
    bool operator!=(NgayThangNam another);
    bool operator>=(NgayThangNam another);
    bool operator<=(NgayThangNam another);
    bool operator>(NgayThangNam another);
    bool operator<(NgayThangNam another);
    friend ostream &operator<<(ostream &out, NgayThangNam &ntn);
    friend istream &operator>>(istream &in, NgayThangNam &ntn);
    int toDays() const;
};

```

Bảng 64. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam

4.1 - Phương thức của class NgayThangNam:

4.1.1 - Phương thức NgayThangNam():

Nội dung: Khởi tạo giá trị ban đầu cho các biến ngày, tháng, năm.

```

284     NgayThangNam::NgayThangNam()
285     {
286         iNgay = 1;
287         iThang = 1;
288         iNam = 1970;
289     }

```

Hình 52. Nội dung của phương thức NgayThangNam()

```

NgayThangNam: :NgayThangNam()
{
    iNgay = 1;
    iThang = 1;
    iNam = 1970;
}

```

*Bảng 65. Nội dung của phương thức NgayThangNam()***4.1.2 - Phương thức NgayThangNam(int Nam, int Thang, int Ngay):**

- Nội dung: Gán giá trị đầu vào cho các biến ngày, tháng, năm.

```

290  NgayThangNam: :NgayThangNam(int Nam, int Thang, int Ngay)
291  {
292      iNam = Nam;
293      iThang = Thang;
294      iNgay = Ngay;
295  }

```

Hình 53. Nội dung của phương thức NgayThangNam(int Nam, int Thang, int Ngay)

```

NgayThangNam: :NgayThangNam(int Nam, int Thang, int Ngay)
{
    iNam = Nam;
    iThang = Thang;
    iNgay = Ngay;
}

```

*Bảng 66. Nội dung của phương thức NgayThangNam(int Nam, int Thang, int Ngay)***4.1.3 - Phương thức TinhNgay():**

- Input: Thời điểm cần đổi sang ngày thứ n.
- Output: Ngày thứ n trong năm.
- Hướng giải quyết:
 1. Tạo mảng chứa các ngày của từng tháng.
 2. Nếu năm đó là năm nhuận thì tháng 2 = 29 ngày.
 3. Ngày thứ n bằng ngày của tháng tại thời điểm đó cộng cho các ngày của các tháng trước.
 4. Trả về ngày thứ n trong năm.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
297 void NgayThangNam::TinhNgay()
298 {
299     int daysInMonth[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
300
301     // Adjust for leap year
302     if (iNam % 4 == 0 && (iNam % 100 != 0 || iNam % 400 == 0))
303     {
304         daysInMonth[1] = 29;
305     }
306
307     int dayOfYear = iNgay;
308
309     for (int i = 0; i < iThang - 1; ++i)
310     {
311         dayOfYear += daysInMonth[i];
312     }
313
314     cout << "Ngày thu: " << dayOfYear << " trong nam" << endl;
315 }
```

Hình 54. Nội dung của phương thức *TinhNgay()*

```
void NgayThangNam::TinhNgay()
{
    int daysInMonth[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

    // Adjust for leap year
    if (iNam % 4 == 0 && (iNam % 100 != 0 || iNam % 400 == 0))
    {
        daysInMonth[1] = 29;
    }

    int dayOfYear = iNgay;

    for (int i = 0; i < iThang - 1; ++i)
    {
        dayOfYear += daysInMonth[i];
    }

    cout << "Ngày thu: " << dayOfYear << " trong nam" << endl;
}
```

Bảng 67. Nội dung của phương thức *TinhNgay()*

Input: 10/4/2024

Output:

Ngày thu: 101 trong năm

Bảng 68. Ví dụ Input/Output của phương thức TinhNgày()

4.1.4 - Phương thức operator+(int ngay):

- Input: n ngày muốn cộng thêm.
- Output: Thời điểm sau khi cộng thêm n ngày.
- Hướng giải quyết:
 1. Cộng ngày hiện tại với n ngày.
 2. Nếu số ngày lớn hơn số ngày trong tháng thì số ngày bằng số ngày hiện tại trừ đi số ngày trong tháng => Tháng ++.
 3. Nếu số Tháng lớn hơn 12 => Năm ++, Tháng reset lại = 1.
 4. Trả về ngày tháng năm sau khi cộng.

```

141  NgayThangNam NgayThangNam::operator+(int ngay)
142  {
143      NgayThangNam temp = *this;
144      temp.iNgay += ngay;
145
146      int daysInMonth[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
147      if (iNam % 4 == 0)
148      {
149          daysInMonth[2] = 29;
150      }
151      while (temp.iNgay > daysInMonth[temp.iThang])
152      {
153          temp.iNgay -= daysInMonth[temp.iThang];
154          temp.iThang++;
155          if (temp.iThang > 12)
156          {
157              temp.iThang = 1;
158              temp.iNam++;
159              if (temp.iNam % 4 == 0)
160              {
161                  daysInMonth[2] = 29;
162              }
163              else
164              {
165                  daysInMonth[2] = 28;
166              }
167          }
168      }
169      return temp;
170  }

```

Hình 55. Nội dung của phương thức operator+(int ngay)

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
NgàyThangNam NgàyThangNam::operator+(int ngay)
{
    NgàyThangNam temp = *this;
    temp.iNgay += ngay;

    int daysInMonth[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    if (iNam % 4 == 0)
    {
        daysInMonth[2] = 29;
    }
    while (temp.iNgay > daysInMonth[temp.iThang])
    {
        temp.iNgay -= daysInMonth[temp.iThang];
        temp.iThang++;
        if (temp.iThang > 12)
        {
            temp.iThang = 1;
            temp.iNam++;
            if (temp.iNam % 4 == 0)
            {
                daysInMonth[2] = 29;
            }
            else
            {
                daysInMonth[2] = 28;
            }
        }
    }
    return temp;
}
```

Bảng 69. Nội dung của phương thức operator+(int ngay)

| |
|------------------------------------|
| Input: 10000 |
| //Giả sử ngày hiện tại là 1/1/2020 |
| Output: 19/5/2047 |

Bảng 70. Ví dụ Input/Output của phương thức operator+(int ngay)

4.1.5 - Phương thức operator-(int ngay):

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Input: n ngày muốn trừ đi.
- Output: Thời điểm sau khi trừ đi n ngày.
- Hướng giải quyết:
 1. Trừ ngày hiện tại với n ngày.
 2. Nếu số ngày bé hơn 1 thì ngày += số ngày trong tháng => Tháng --
 3. Nếu số Tháng bé hơn 1 => Năm --, Tháng reset lại = 12.
 4. Trả về ngày tháng năm sau khi trừ.

```
110     NgayThangNam NgayThangNam::operator-(int ngay)
111     {
112         NgayThangNam temp = *this;
113         temp.iNgay -= ngay;
114
115         int daysInMonth[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
116         if (temp.iNam % 4 == 0)
117         {
118             daysInMonth[2] = 29;
119         }
120         while (temp.iNgay < 1)
121         {
122             temp.iNgay += daysInMonth[temp.iThang - 1];
123             temp.iThang--;
124             if (temp.iThang < 1)
125             {
126                 temp.iThang = 12;
127                 temp.iNam--;
128                 if (temp.iNam % 4 == 0)
129                 {
130                     daysInMonth[2] = 29;
131                 }
132                 else
133                 {
134                     daysInMonth[2] = 28;
135                 }
136             }
137         }
138         return temp;
139     }
```

Hình 56. Nội dung của phương thức operator-(int ngay)

```
NgayThangNam NgayThangNam::operator-(int ngay)
{
    NgayThangNam temp = *this;
    temp.iNgay -= ngay;

    int daysInMonth[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
```

```

    if (temp.iNam % 4 == 0)
    {
        daysInMonth[2] = 29;
    }
    while (temp.iNgày < 1)
    {
        temp.iNgày += daysInMonth[temp.iThang - 1];
        temp.iThang--;
        if (temp.iThang < 1)
        {
            temp.iThang = 12;
            temp.iNam--;
            if (temp.iNam % 4 == 0)
            {
                daysInMonth[2] = 29;
            }
            else
            {
                daysInMonth[2] = 28;
            }
        }
    }
    return temp;
}

```

Bảng 71. Nội dung của phương thức operator-(int ngay)

| |
|---|
| Input: 10000 //Giả sử thời điểm hiện tại là 1/1/2020 |
| Output: 28/1/1990 |

*Bảng 72. Nội dung của phương thức operator-(int ngay)***4.1.6 - Phương thức toDays() const:**

- Nội dung: Dùng để hỗ trợ cho phương thức operator-(const NgayThangNam &a) const. Đây là phương thức đếm số ngày từ một thời điểm cố định đến thời điểm hiện tại.
- Hướng giải quyết:

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

1. Tạo 1 biến ngày tạm = 0.
2. Chạy vòng lặp for cho một thời gian cố định giả sử năm 1900, giá trị này tăng dần.
 - 2.1. Nếu 1900 < hơn năm hiện tại thì biến ngày tạm += số ngày trong năm kể từ 1900.
3. Chạy vòng lặp for với giá trị tháng = 1, giá trị này tăng dần.
 - 3.1. Nếu 1 < tháng hiện tại thì biến ngày tạm += số ngày trong các tháng trước đó.
4. Biến ngày tạm += biến ngày hiện tại.

***Lưu ý:** Nếu thời gian hiện tại trước thời điểm cố định thì chương trình sẽ phát sinh lỗi => cần điều chỉnh lại thời gian cố định.

```
85  int NgayThangNam::toDays() const
86  {
87      int daysInMonth[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
88      if (iNam % 4 == 0 && (iNam % 100 != 0 || iNam % 400 == 0))
89      {
90          daysInMonth[2] = 29;
91      }
92
93      int days = 0;
94      for (int year = 1900; year < iNam; year++)
95      {
96          days += year % 4 == 0 && (year % 100 != 0 || year % 400 == 0) ? 366
97              : 365;
98      }
99      for (int month = 1; month < iThang; month++)
100     {
101         days += daysInMonth[month];
102     }
103     days += iNgay;
104     return days;
105 }
```

Hình 57. Nội dung của phương thức toDays() const

```
int NgayThangNam::toDays() const
{
    int daysInMonth[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    if (iNam % 4 == 0 && (iNam % 100 != 0 || iNam % 400 == 0))
    {
        daysInMonth[2] = 29;
    }
}
```

```

    }

    int days = 0;
    for (int year = 1900; year < iNam; year++)
    {
        days += year % 4 == 0 && (year % 100 != 0 || year % 400 == 0) ? 366
: 365;
    }
    for (int month = 1; month < iThang; month++)
    {
        days += daysInMonth[month];
    }
    days += iNgay;

    return days;
}

```

Bảng 73. Nội dung của phương thức toDays() const

| |
|-----------------|
| Input: 1/1/1960 |
| Output: 10985 |

Bảng 74. Ví dụ Input/Output của phương thức toDays() const

4.1.7 - Phương thức operator-(const NgayThangNam &a) const:

- Input: Số ngày kể từ ngày cố định của phương thức **toDays() const**.
- Output: Khoảng cách giữa 2 thời điểm.
- Hướng giải quyết: Số ngày hiện tại kể từ thời điểm cố định trừ đi số ngày kể từ thời điểm cố định của thời điểm muốn xét.

```

106 int NgayThangNam::operator-(const NgayThangNam &a) const
107 {
108     return toDays() - a.toDays();
109 }

```

Hình 58. Nội dung của phương thức operator-(const NgayThangNam &a) const

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
int NgayThangNam::operator-(const NgayThangNam &a) const
{
    return toDays() - a.toDays();
}
```

Bảng 75. Nội dung của phương thức operator-(const NgayThangNam &a) const

Input: 28/1/1930
//Giả sử ngày hiện tại là 10/4/2024

Output: 34406

Bảng 76. Ví dụ Input/Output của phương thức operator-(const NgayThangNam &a) const

4.1.8 - Phương thức operator++(int):

- Nội dung: Thời gian sau khi cộng thêm 1 ngày.

```

62 NgayThangNam NgayThangNam::operator++(int)
63 {
64
65     iNgay++;
66     if ((iNgay > 30 && (iThang == 4 || iThang == 6 || iThang == 9 || iThang
67 == 11)) ||
68         (iNgay > 31 && (iThang == 1 || iThang == 3 || iThang == 5 || iThang
69 == 7 || iThang == 8 || iThang == 10 || iThang == 12)) ||
70         (iNgay > 28 && iThang == 2 && iNam % 4 != 0) || (iNgay > 29 &&
71 iThang == 2 && iNam % 4 == 0))
72     {
73         iNgay = 1;
74         iThang++;
75         if (iThang > 12)
76         {
77             iThang = 1;
78             iNam++;
79         }
80     }
81
82     return *this;
83 }

```

Hình 59. Nội dung của phương thức `operator++(int)`

```

NgàyThangNam NgàyThangNam::operator++(int)
{
    iNgày++;
}

```


IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
if ((iNgay > 30 && (iThang == 4 || iThang == 6 || iThang == 9 || iThang == 11)) ||  
    (iNgay > 31 && (iThang == 1 || iThang == 3 || iThang == 5 || iThang == 7 || iThang == 8 || iThang == 10 || iThang == 12)) ||  
    (iNgay > 28 && iThang == 2 && iNam % 4 != 0) || (iNgay > 29 &&  
iThang == 2 && iNam % 4 == 0))  
{  
    iNgay = 1;  
    iThang++;  
    if (iThang > 12)  
    {  
        iThang = 1;  
        iNam++;  
    }  
}  
  
return *this;  
}
```

Bảng 77. Nội dung của phương thức operator++(int)

| |
|-------------------|
| Input: 10/4/2024 |
| Output: 11/4/2024 |

Bảng 78. Ví dụ Input/Output của phương thức operator++(int)

4.1.9 - Phương thức operator--(int):

- Nội dung: Thời gian sau khi cộng thêm 1 ngày.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
29  ~ NgayThangNam NgayThangNam::operator--(int)
30  {
31  |
32      iNgay--;
33  ~  if (iNgay < 1)
34      {
35  ~      iThang--;
36  ~      if (iThang < 1)
37          {
38              iThang = 12;
39              iNam--;
40          }
41  ~      if (iThang == 2 && iNam % 4 == 0)
42          {
43              iNgay = 29;
44          }
45  ~      else if (iThang == 2 && iNam % 4 != 0)
46          {
47              iNgay = 28;
48          }
49  ~      else if (iThang == 1 || iThang == 3 || iThang == 5 || iThang == 7 ||
50              iThang == 8 || iThang == 10 || iThang == 12)
51          {
52              iNgay = 31;
53          }
54  ~      else
55          {
56              iNgay = 30;
57          }
58  |
59      return *this;
60  }
```

Hình 60. Nội dung của phương thức operator--(int)

```
NgayThangNam NgayThangNam::operator--(int)
{

    iNgay--;
    if (iNgay < 1)
    {
        iThang--;
        if (iThang < 1)
        {
            iThang = 12;
            iNam--;
        }
        if (iThang == 2 && iNam % 4 == 0)
```

```

    {
        iNgày = 29;
    }
    else if (iThang == 2 && iNam % 4 != 0)
    {
        iNgày = 28;
    }
    else if (iThang == 1 || iThang == 3 || iThang == 5 || iThang == 7 ||
iThang == 8 || iThang == 10 || iThang == 12)
    {
        iNgày = 31;
    }
    else
    {
        iNgày = 30;
    }
}

return *this;
}

```

Bảng 79. Nội dung của phương thức operator--(int)

| |
|-------------------|
| Input: 11/4/2024 |
| Output: 10/4/2024 |

Bảng 80. Ví dụ Input/Output của phương thức operator--(int)

4.1.10 - Phương thức operator==(NgàyThangNam another):

- Nội dung: So sánh bằng giữa 2 thời điểm

```

169  bool NgàyThangNam::operator==(NgàyThangNam another)
170  {
171      return iNgày == another.iNgày && iThang == another.iThang && iNam ==
        another.iNam;
172  }

```

Hình 61. Nội dung của phương thức operator==(NgàyThangNam another)

```
bool NgayThangNam::operator==(NgayThangNam another)
{
    return iNgay == another.iNgay && iThang == another.iThang && iNam ==
another.iNam;
}
```

Bảng 81. Nội dung của phương thức operator==(NgayThangNam another)

4.1.11 - Phương thức operator!=(NgayThangNam another):

- Nội dung: So sánh không bằng giữa 2 thời điểm

```
174 bool NgayThangNam::operator!=(NgayThangNam another)
175 {
176     return !(*this == another);
177 }
```

Hình 62. Nội dung của phương thức operator!=(NgayThangNam another)

```
bool NgayThangNam::operator!=(NgayThangNam another)
{
    return !(*this == another);
}
```

Bảng 82. Nội dung của phương thức operator!=(NgayThangNam another)

4.1.12 - Phương thức operator>=(NgayThangNam another):

- Nội dung: So sánh lớn hơn bằng, nếu thời điểm hiện tại lớn hơn hoặc bằng thời điểm **another** => return **true**.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
245 bool NgayThangNam::operator>=(NgayThangNam another)
246 {
247     if (this->iNam > another.iNam)
248         return true;
249     else if (this->iNam == another.iNam)
250     {
251         if (this->iThang > another.iThang)
252             return true;
253         else if (this->iThang == another.iThang)
254         {
255             if (this->iNgay >= another.iNgay)
256                 return true;
257             else
258                 return false;
259         }
260         else
261             return false;
262     }
263     else
264         return false;
265 }
```

Hình 63. Nội dung của phương thức `operator>=(NgayThangNam another)`

```
bool NgayThangNam::operator>=(NgayThangNam another)
{
    if (this->iNam > another.iNam)
        return true;
    else if (this->iNam == another.iNam)
    {
        if (this->iThang > another.iThang)
            return true;
        else if (this->iThang == another.iThang)
        {
            if (this->iNgay >= another.iNgay)
                return true;
            else
                return false;
        }
    }
}
```

```

        else
            return false;
    }
    else
        return false;
}

```

Bảng 83. Nội dung của phương thức `operator>=(NgayThangNam another)`

4.1.13 - Phương thức `operator<=(NgayThangNam another)`:

- Nội dung: So sánh bé hơn bằng, nếu thời điểm hiện tại bé hơn hoặc bằng thời điểm **another** => return **true**.

```

223  bool NgayThangNam::operator<=(NgayThangNam another)
224  {
225      if (this->iNam < another.iNam)
226          return true;
227      else if (this->iNam == another.iNam)
228      {
229          if (this->iThang < another.iThang)
230              return true;
231          else if (this->iThang == another.iThang)
232          {
233              if (this->iNgay <= another.iNgay)
234                  return true;
235              else
236                  return false;
237          }
238          else
239              return false;
240      }
241      else
242          return false;
243  }

```

Hình 64. Nội dung của phương thức `operator<=(NgayThangNam another)`

```

bool NgayThangNam::operator<=(NgayThangNam another)
{
    if (this->iNam < another.iNam)

```

```
        return true;
    else if (this->iNam == another.iNam)
    {
        if (this->iThang < another.iThang)
            return true;
        else if (this->iThang == another.iThang)
        {
            if (this->iNgay <= another.iNgay)
                return true;
            else
                return false;
        }
        else
            return false;
    }
    else
        return false;
}
```

Bảng 84. Nội dung của phương thức operator<=(NgàyThangNam another)

4.1.14 - Phương thức operator>(NgàyThangNam another):

- Nội dung: So sánh lớn hơn , nếu thời điểm hiện tại lớn hơn thời điểm **another** => return **true**.

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

```
201  bool NgayThangNam::operator>(NgayThangNam another)
202  {
203      if (this->iNam > another.iNam)
204          return true;
205      else if (this->iNam == another.iNam)
206      {
207          if (this->iThang > another.iThang)
208              return true;
209          else if (this->iThang == another.iThang)
210          {
211              if (this->iNgay > another.iNgay)
212                  return true;
213              else
214                  return false;
215          }
216          else
217              return false;
218      }
219      else
220          return false;
221  }
```

Hình 65. Nội dung của phương thức operator>(NgayThangNam another)

```
bool NgayThangNam::operator>(NgayThangNam another)
{
    if (this->iNam > another.iNam)
        return true;
    else if (this->iNam == another.iNam)
    {
        if (this->iThang > another.iThang)
            return true;
        else if (this->iThang == another.iThang)
        {
            if (this->iNgay > another.iNgay)
                return true;
            else
                return false;
        }
    }
}
```



```

        else
            return false;
    }
    else
        return false;
}

```

Bảng 85. Nội dung của phương thức operator>(NgàyThangNam another)

4.1.15 - Phương thức operator<(NgàyThangNam another):

- Nội dung: So sánh bé hơn , nếu thời điểm hiện tại bé hơn thời điểm **another** => return **true**.

```

179  bool NgàyThangNam::operator<(NgàyThangNam another)
180  {
181      if (this->iNam < another.iNam)
182          return true;
183      else if (this->iNam == another.iNam)
184      {
185          if (this->iThang < another.iThang)
186              return true;
187          else if (this->iThang == another.iThang)
188          {
189              if (this->iNgày < another.iNgày)
190                  return true;
191              else
192                  return false;
193          }
194          else
195              return false;
196      }
197      else
198          return false;
199  }

```

Hình 66. Nội dung của phương thức operator<(NgàyThangNam another)

```

bool NgàyThangNam::operator<(NgàyThangNam another)
{
    if (this->iNam < another.iNam)
        return true;
}

```

```

else if (this->iNam == another.iNam)
{
    if (this->iThang < another.iThang)
        return true;
    else if (this->iThang == another.iThang)
    {
        if (this->iNgay < another.iNgay)
            return true;
        else
            return false;
    }
    else
        return false;
}
else
    return false;
}

```

*Bảng 86. Nội dung của phương thức operator<(NgàyThangNam another)***4.1.16 - Phương thức operator<<(ostream &out, NgàyThangNam &ntn):**

- Nội dung: Overload toán tử xuất <<.

```

276  ostream &operator<<(ostream &out, NgàyThangNam &ntn)
277  {
278  |||      out << ntn.iNgay << "/" << ntn.iThang << "/" << ntn.iNam;
279  |      return out;
280  }

```

Hình 67. Nội dung của phương thức operator<<(ostream &out, NgàyThangNam &ntn)

```

ostream &operator<<(ostream &out, NgàyThangNam &ntn)
{
    out << ntn.iNgay << "/" << ntn.iThang << "/" << ntn.iNam;
    return out;
}

```

*Bảng 87. Nội dung của phương thức operator<<(ostream &out, NgàyThangNam &ntn)***4.1.17 - Phương thức operator>>(ostream &out, NgàyThangNam &ntn):**

IT002 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Nội dung: Overload toán tử nhập >>.

```
266  ✓ istream &operator>>(istream &in, NgayThangNam &ntn)
267  {
268      cout << "Nhap ngay: ";
269      in >> ntn.iNgay;
270      cout << "Nhap thang: ";
271      in >> ntn.iThang;
272      cout << "Nhap nam: ";
273      in >> ntn.iNam;
274      return in;
275  }
```

Hình 68. Nội dung của phương thức `operator>>(ostream &out, NgayThangNam &ntn)`

```
istream &operator>>(istream &in, NgayThangNam &ntn)
{
    cout << "Nhap ngay: ";
    in >> ntn.iNgay;
    cout << "Nhap thang: ";
    in >> ntn.iThang;
    cout << "Nhap nam: ";
    in >> ntn.iNam;
    return in;
}
```

Bảng 88. Nội dung của phương thức `operator>>(ostream &out, NgayThangNam &ntn)`

Link Google Drive:

<https://drive.google.com/drive/folders/1Ky7spqSf57jnscwO6tjRJhLQDJOF9zZE?usp=sharing>