**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 Quí

Sinh viên thực hiện: Nguyễn Thanh Tùng

🙡🙢 Tp. Hồ Chí Minh, 04/2024 🙠🙣

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2024*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**Mục Lục**

[Bài tập 1: Xây dựng lớp phân số: 10](#_Toc164625782)

[1.1 – Phương thức của class PhanSo 12](#_Toc164625783)

[1.1.1 – Phương thức PhanSo(): 12](#_Toc164625784)

[1.1.2 – Phương thức PhanSo(int Tu, int Mau): 12](#_Toc164625785)

[1.1.3 – Phương thức RutGon(int &Tu, int &Mau): 13](#_Toc164625786)

[1.1.4 – Phương thức operator+(PhanSo another): 14](#_Toc164625787)

[1.1.5 – Phương thức operator-(PhanSo another): 16](#_Toc164625788)

[1.1.6 – Phương thức operator\*(PhanSo another): 18](#_Toc164625789)

[1.1.7 – Phương thức operator/(PhanSo another): 20](#_Toc164625790)

[1.1.8 – Phương thức operator==(PhanSo another): 22](#_Toc164625791)

[1.1.9 – Phương thức operator!=(PhanSo another): 22](#_Toc164625792)

[1.1.10 – Phương thức operator>=(PhanSo another): 23](#_Toc164625793)

[1.1.11 – Phương thức operator<=(PhanSo another): 23](#_Toc164625794)

[1.1.12 – Phương thức operator>(PhanSo another): 24](#_Toc164625795)

[1.1.13 – Phương thức operator<(PhanSo another): 25](#_Toc164625796)

[1.1.14 – Phương thức operator>>(istream &in, PhanSo &ps): 25](#_Toc164625797)

[1.1.15 – Phương thức operator<<(ostream &out, PhanSo &ps): 26](#_Toc164625798)

[Bài tập 2: Xây dựng lớp số phức: 27](#_Toc164625799)

[2.1 – Phương thức của class SoPhuc: 29](#_Toc164625800)

[2.1.1 – Phương thức SoPhuc() 29](#_Toc164625801)

[2.1.2 – Phương thức SoPhuc(int thuc, int ao) 29](#_Toc164625802)

[2.1.3 – Phương thức operator+(SoPhuc another) 30](#_Toc164625803)

[2.1.4 – Phương thức operator-(SoPhuc another) 31](#_Toc164625804)

[2.1.5 – Phương thức operator\*(SoPhuc another) 31](#_Toc164625805)

[2.1.6 – Phương thức operator/(SoPhuc another) 32](#_Toc164625806)

[2.1.7 – Phương thức operator==(SoPhuc another) 33](#_Toc164625807)

[2.1.8 – Phương thức operator!=(SoPhuc another) 34](#_Toc164625808)

[2.1.9 – Phương thức operator>>(istream &in, SoPhuc &another) 34](#_Toc164625809)

[2.1.10 – Phương thức operator<<(ostream &out, SoPhuc &another) 35](#_Toc164625810)

[Bài tập 3. Xây dựng lớp thời gian 35](#_Toc164625811)

[3.1 – Phương thức của class ThoiGian: 38](#_Toc164625812)

[3.1.1 – Phương thức ThoiGian(): 38](#_Toc164625813)

[3.1.2 – Phương thức ThoiGian(int Gio, int Phut, int Giay): 39](#_Toc164625814)

[3.1.3 – Phương thức TinhGiay(): 39](#_Toc164625815)

[3.1.4 – Phương thức TinhLaiGio(): 40](#_Toc164625816)

[3.1.5 – Phương thức operator+(int Giay): 41](#_Toc164625817)

[3.1.6 – Phương thức operator-(int Giay): 43](#_Toc164625818)

[3.1.7 – Phương thức operator+(ThoiGian a): 45](#_Toc164625819)

[3.1.8 – Phương thức operator-(ThoiGian a): 47](#_Toc164625820)

[3.1.9 – Phương thức operator++(int): 49](#_Toc164625821)

[3.1.10 – Phương thức operator--(int): 52](#_Toc164625822)

[3.1.11 – Phương thức operator==(ThoiGian another): 55](#_Toc164625823)

[3.1.12 - Phương thức operator!=(ThoiGian another): 55](#_Toc164625824)

[3.1.13 - Phương thức operator>=(ThoiGian another): 55](#_Toc164625825)

[3.1.14 - Phương thức operator<=(ThoiGian another): 56](#_Toc164625826)

[3.1.15 - Phương thức operator>(ThoiGian another): 56](#_Toc164625827)

[3.1.16 - Phương thức operator<(ThoiGian another): 57](#_Toc164625828)

[3.1.17 - Phương thức operator>>(istream &in, ThoiGian &b): 57](#_Toc164625829)

[3.1.18 - Phương thức operator<<(ostream &out, ThoiGian &b): 58](#_Toc164625830)

[Bài tập 4. Xây dựng lớp ngày tháng năm 59](#_Toc164625831)

[4.1 - Phương thức của class NgayThangNam: 61](#_Toc164625832)

[4.1.1 - Phương thức NgayThangNam(): 61](#_Toc164625833)

[4.1.2 - Phương thức NgayThangNam(int Nam, int Thang, int Ngay): 62](#_Toc164625834)

[4.1.3 - Phương thức TinhNgay(): 62](#_Toc164625835)

[4.1.4 - Phương thức operator+(int ngay): 64](#_Toc164625836)

[4.1.5 - Phương thức operator-(int ngay): 65](#_Toc164625837)

[4.1.6 - Phương thức toDays() const: 67](#_Toc164625838)

[4.1.7 - Phương thức operator-(const NgayThangNam &a) const: 69](#_Toc164625839)

[4.1.8 - Phương thức operator++(int): 70](#_Toc164625840)

[4.1.9 - Phương thức operator--(int): 71](#_Toc164625841)

[4.1.10 - Phương thức operator==(NgayThangNam another): 73](#_Toc164625842)

[4.1.11 - Phương thức operator!=(NgayThangNam another): 74](#_Toc164625843)

[4.1.12 - Phương thức operator>=(NgayThangNam another): 74](#_Toc164625844)

[4.1.13 - Phương thức operator<=(NgayThangNam another): 76](#_Toc164625845)

[4.1.14 - Phương thức operator>(NgayThangNam another): 77](#_Toc164625846)

[4.1.15 - Phương thức operator<(NgayThangNam another): 79](#_Toc164625847)

[4.1.16 - Phương thức operator<<(ostream &out, NgayThangNam &ntn): 80](#_Toc164625848)

[4.1.17 - Phương thức operator>>(ostream &out, NgayThangNam &ntn): 80](#_Toc164625849)

**DANH MỤC BẢNG**

[Bảng 1. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo 13](#_Toc164625850)

[Bảng 2. Nội dung của phương thức PhanSo() 13](#_Toc164625851)

[Bảng 3. Nội dung của phương thức PhanSo(int Tu, int Mau) 14](#_Toc164625852)

[Bảng 4. Nội dung của phương thức RutGon(int &Tu, int &Mau) 15](#_Toc164625853)

[Bảng 5. Ví dụ về Input/Output của phương thức RutGon(int &Tu, int &Mau) 15](#_Toc164625854)

[Bảng 6. Nội dung của phương thức operator+(PhanSo another) 17](#_Toc164625855)

[Bảng 7. Ví dụ Input/Output của phương thức operator+(PhanSo another) 17](#_Toc164625856)

[Bảng 8. Nội dung của phương thức operator-(PhanSo another) 19](#_Toc164625857)

[Bảng 9. Ví dụ Input/Output của phương thức operator-(PhanSo another) 19](#_Toc164625858)

[Bảng 10. Nội dung phương thức operator\*(PhanSo another) 21](#_Toc164625859)

[Bảng 11. Ví dụ về Input/Output của phương thức operator\*(PhanSo another) 21](#_Toc164625860)

[Bảng 12. Nội dung phương thức operator/(PhanSo another) 23](#_Toc164625861)

[Bảng 13. Ví dụ về Input/Output của phương thức operator/(PhanSo another) 23](#_Toc164625862)

[Bảng 14. Nội dung của phương thức operator==(PhanSo another) 23](#_Toc164625863)

[Bảng 15. Nội dung của phương thức operator!=(PhanSo another) 24](#_Toc164625864)

[Bảng 16. Nội dung của phương thức operator>=(PhanSo another) 24](#_Toc164625865)

[Bảng 17. Nội dung của phương thức operator<=(PhanSo another) 25](#_Toc164625866)

[Bảng 18. Nội dung của phương thức operator>(PhanSo another) 26](#_Toc164625867)

[Bảng 19. Nội dung của phương thức operator<(PhanSo another) 26](#_Toc164625868)

[Bảng 20. Nội dung của phương thức operator>>(istream &in, PhanSo &ps) 27](#_Toc164625869)

[Bảng 21. Nội dung của phương thức operator<<(istream &in, PhanSo &ps) 27](#_Toc164625870)

[Bảng 22. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc 30](#_Toc164625871)

[Bảng 23. Nội dung của phương thức SoPhuc() 30](#_Toc164625872)

[Bảng 24. Nội dung của phương thức SoPhuc(int thuc, int ao) 31](#_Toc164625873)

[Bảng 25. Nội dung của phương thức operator+(SoPhuc another) 31](#_Toc164625874)

[Bảng 26. Ví dụ Input/Output của phương thức operator+(SoPhuc another) 31](#_Toc164625875)

[Bảng 27. Nội dung của phương thức operator-(SoPhuc another) 32](#_Toc164625876)

[Bảng 28. Ví dụ về Input/Output của phương thức operator-(SoPhuc another) 32](#_Toc164625877)

[Bảng 29. Nội dung của phương thức operator\*(SoPhuc another) 33](#_Toc164625878)

[Bảng 30. Ví dụ Input/Output của phương thức operator\*(SoPhuc another) 33](#_Toc164625879)

[Bảng 31. Nội dung của phương thức operator/(SoPhuc another) 34](#_Toc164625880)

[Bảng 32. Ví dụ Input/Output của phương thức operator/(SoPhuc another) 34](#_Toc164625881)

[Bảng 33. Nội dung của phương thức operator==(SoPhuc another) 35](#_Toc164625882)

[Bảng 34. Nội dung của phương thức operator!=(SoPhuc another) 35](#_Toc164625883)

[Bảng 35. Nội dung phương thức operator>>(istream &in, SoPhuc &another) 36](#_Toc164625884)

[Bảng 36. Nội dung của phương thức operator<<(ostream &out, SoPhuc &another) 36](#_Toc164625885)

[Bảng 37. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian 39](#_Toc164625886)

[Bảng 38. Nội dung của phương thức ThoiGian() 40](#_Toc164625887)

[Bảng 39. Nội dung của phương thức ThoiGian(int Gio, int Phut, int Giay) 40](#_Toc164625888)

[Bảng 40. Nội dung của phương thức TinhGiay() 41](#_Toc164625889)

[Bảng 41. Ví dụ Input/Output của phương thức TinhGiay() 41](#_Toc164625890)

[Bảng 42. Nội dung của phương thức TinhLaiGio(). 42](#_Toc164625891)

[Bảng 43. Ví dụ Input/Output của phương thức TinhLaiGio(). 42](#_Toc164625892)

[Bảng 44. Nội dung của phương thức operator+(int Giay) 44](#_Toc164625893)

[Bảng 45. Ví dụ Input/Output của phương thức operator+(int Giay) 44](#_Toc164625894)

[Bảng 46. Nội dung của phương thức operator-(int Giay) 46](#_Toc164625895)

[Bảng 47. Ví dụ Input/Output của phương thức operator-(int Giay) 46](#_Toc164625896)

[Bảng 48. Nội dung của phương thức operator+(ThoiGian a) 48](#_Toc164625897)

[Bảng 49. Ví dụ Input/Output của phương thức operator+(ThoiGian a) 48](#_Toc164625898)

[Bảng 50. Nội dung của phương thức operator-(ThoiGian a) 50](#_Toc164625899)

[Bảng 51. Ví dụ Input/Output của phương thức operator-(ThoiGian a) 50](#_Toc164625900)

[Bảng 52. Nội dung phương thức operator++(int) 52](#_Toc164625901)

[Bảng 53. Ví dụ Input/Output của phương thức operator++(int) 53](#_Toc164625902)

[Bảng 54. Nội dung của phương thức operator--(int) 55](#_Toc164625903)

[Bảng 55. Ví dụ Input/Output của phương thức operator--(int) 56](#_Toc164625904)

[Bảng 56. Nội dung của phương thức operator==(ThoiGian another) 56](#_Toc164625905)

[Bảng 57. Nội dung của phương thức operator!=(ThoiGian another) 56](#_Toc164625906)

[Bảng 58. Nội dung của phương thức operator>=(ThoiGian another) 57](#_Toc164625907)

[Bảng 59. Nội dung của phương thức operator>=(ThoiGian another) 57](#_Toc164625908)

[Bảng 60. Nội dung của phương thức operator>(ThoiGian another) 58](#_Toc164625909)

[Bảng 61. Nội dung của phương thức operator<(ThoiGian another) 58](#_Toc164625910)

[Bảng 62. Nội dung của phương thức operator>>(istream &in, ThoiGian &b) 59](#_Toc164625911)

[Bảng 63. Nội dung của phương thức operator<<(ostream &out, ThoiGian &b) 60](#_Toc164625912)

[Bảng 64. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam 62](#_Toc164625913)

[Bảng 65. Nội dung của phương thức NgayThangNam() 63](#_Toc164625914)

[Bảng 66. Nội dung của phương thức NgayThangNam(int Nam, int Thang, int Ngay) 63](#_Toc164625915)

[Bảng 67. Nội dung của phương thức TinhNgay() 64](#_Toc164625916)

[Bảng 68. Ví dụ Input/Output của phương thức TinhNgay() 65](#_Toc164625917)

[Bảng 69. Nội dung của phương thức operator+(int ngay) 66](#_Toc164625918)

[Bảng 70. Ví dụ Input/Output của phương thức operator+(int ngay) 66](#_Toc164625919)

[Bảng 71. Nội dung của phương thức operator-(int ngay) 68](#_Toc164625920)

[Bảng 72. Nội dung của phương thức operator-(int ngay) 68](#_Toc164625921)

[Bảng 73. Nội dung của phương thức toDays() const 70](#_Toc164625922)

[Bảng 74. Ví dụ Input/Output của phương thức toDays() const 70](#_Toc164625923)

[Bảng 75. Nội dung của phương thức operator-(const NgayThangNam &a) const 71](#_Toc164625924)

[Bảng 76. Ví dụ Input/Output của phương thức operator-(const NgayThangNam &a) const 71](#_Toc164625925)

[Bảng 77. Nội dung của phương thức operator++(int) 72](#_Toc164625926)

[Bảng 78. Ví dụ Input/Output của phương thức operator++(int) 72](#_Toc164625927)

[Bảng 79. Nội dung của phương thức operator--(int) 74](#_Toc164625928)

[Bảng 80. Ví dụ Input/Output của phương thức operator--(int) 74](#_Toc164625929)

[Bảng 81. Nội dung của phương thức operator==(NgayThangNam another) 75](#_Toc164625930)

[Bảng 82. Nội dung của phương thức operator!=(NgayThangNam another) 75](#_Toc164625931)

[Bảng 83. Nội dung của phương thức operator>=(NgayThangNam another) 77](#_Toc164625932)

[Bảng 84. Nội dung của phương thức operator<=(NgayThangNam another) 78](#_Toc164625933)

[Bảng 85. Nội dung của phương thức operator>(NgayThangNam another) 80](#_Toc164625934)

[Bảng 86. Nội dung của phương thức operator<(NgayThangNam another) 81](#_Toc164625935)

[Bảng 87. Nội dung của phương thức operator<<(ostream &out, NgayThangNam &ntn) 81](#_Toc164625936)

[Bảng 88. Nội dung của phương thức operator>>(ostream &out, NgayThangNam &ntn) 82](#_Toc164625937)

**DANH MỤC HÌNH ẢNH**

[Hình 1. Class diagram của lớp PhanSo 11](#_Toc164625938)

[Hình 2. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp PhanSo 12](#_Toc164625939)

[Hình 3. Nội dung của phương thức PhanSo() 13](#_Toc164625940)

[Hình 4. Nội dung của phương thức PhanSo(int Tu, int Mau) 14](#_Toc164625941)

[Hình 5. Nội dung của phương thức RutGon(int &Tu, int &Mau) 14](#_Toc164625942)

[Hình 6. Nội dung của phương thức operator+(PhanSo another) 16](#_Toc164625943)

[Hình 7. Nội dung của phương thức operator-(PhanSo another) 18](#_Toc164625944)

[Hình 8. Nội dung phương thức operator\*(PhanSo another) 20](#_Toc164625945)

[Hình 9. Nội dung phương thức operator/(PhanSo another) 22](#_Toc164625946)

[Hình 10. Nội dung của phương thức operator==(PhanSo another) 23](#_Toc164625947)

[Hình 11. Nội dung của phương thức operator!=(PhanSo another) 24](#_Toc164625948)

[Hình 12. Nội dung của phương thức operator>=(PhanSo another) 24](#_Toc164625949)

[Hình 13. Nội dung của phương thức operator<=(PhanSo another) 25](#_Toc164625950)

[Hình 14. Nội dung của phương thức operator>(PhanSo another) 25](#_Toc164625951)

[Hình 15. Nội dung của phương thức operator<(PhanSo another) 26](#_Toc164625952)

[Hình 16. Nội dung của phương thức operator>>(istream &in, PhanSo &ps) 27](#_Toc164625953)

[Hình 17. Nội dung của phương thức operator<<(istream &in, PhanSo &ps) 27](#_Toc164625954)

[Hình 18. Class diagram của lớp SoPhuc 28](#_Toc164625955)

[Hình 19. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp SoPhuc 29](#_Toc164625956)

[Hình 20. Nội dung của phương thức SoPhuc() 30](#_Toc164625957)

[Hình 21. Nội dung của phương thức SoPhuc(int thuc, int ao) 30](#_Toc164625958)

[Hình 22. Nội dung của phương thức operator+(SoPhuc another) 31](#_Toc164625959)

[Hình 23. Nội dung của phương thức operator-(SoPhuc another) 32](#_Toc164625960)

[Hình 24. Nội dung của phương thức operator\*(SoPhuc another) 33](#_Toc164625961)

[Hình 25. Nội dung của phương thức operator/(SoPhuc another) 34](#_Toc164625962)

[Hình 26. Nội dung của phương thức operator==(SoPhuc another) 35](#_Toc164625963)

[Hình 27. Nội dung của phương thức operator!=(SoPhuc another) 35](#_Toc164625964)

[Hình 28. Nội dung phương thức operator>>(istream &in, SoPhuc &another) 35](#_Toc164625965)

[Hình 29. Nội dung của phương thức operator<<(ostream &out, SoPhuc &another) 36](#_Toc164625966)

[Hình 30. Class diagram của lớp ThoiGian 37](#_Toc164625967)

[Hình 31. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp ThoiGian 38](#_Toc164625968)

[Hình 32. Nội dung của phương thức ThoiGian() 39](#_Toc164625969)

[Hình 33. Nội dung của phương thức ThoiGian(int Gio, int Phut, int Giay) 40](#_Toc164625970)

[Hình 34. Nội dung của phương thức TinhGiay() 41](#_Toc164625971)

[Hình 35. Nội dung của phương thức TinhLaiGio(). 41](#_Toc164625972)

[Hình 36. Nội dung của phương thức operator+(int Giay) 43](#_Toc164625973)

[Hình 37. Nội dung của phương thức operator-(int Giay) 45](#_Toc164625974)

[Hình 38. Nội dung của phương thức operator+(ThoiGian a) 47](#_Toc164625975)

[Hình 39. Nội dung của phương thức operator-(ThoiGian a) 49](#_Toc164625976)

[Hình 40. Nội dung phương thức operator++(int) 51](#_Toc164625977)

[Hình 41. Nội dung của phương thức operator--(int) 54](#_Toc164625978)

[Hình 42. Nội dung của phương thức operator==(ThoiGian another) 56](#_Toc164625979)

[Hình 43. Nội dung của phương thức operator!=(ThoiGian another) 56](#_Toc164625980)

[Hình 44. Nội dung của phương thức operator>=(ThoiGian another) 57](#_Toc164625981)

[Hình 45. Nội dung của phương thức operator>=(ThoiGian another) 57](#_Toc164625982)

[Hình 46. Nội dung của phương thức operator>(ThoiGian another) 58](#_Toc164625983)

[Hình 47. Nội dung của phương thức operator<(ThoiGian another) 58](#_Toc164625984)

[Hình 48. Nội dung của phương thức operator>>(istream &in, ThoiGian &b) 59](#_Toc164625985)

[Hình 49. Nội dung của phương thức operator<<(ostream &out, ThoiGian &b) 59](#_Toc164625986)

[Hình 50. Class diagram của lớp NgayThangNam 60](#_Toc164625987)

[Hình 51. Thực hiện xây dựng lớp, khai báo thuộc tính, phương thức của lớp NgayThangNam 61](#_Toc164625988)

[Hình 52. Nội dung của phương thức NgayThangNam() 62](#_Toc164625989)

[Hình 53. Nội dung của phương thức NgayThangNam(int Nam, int Thang, int Ngay) 63](#_Toc164625990)

[Hình 54. Nội dung của phương thức TinhNgay() 64](#_Toc164625991)

[Hình 55. Nội dung của phương thức operator+(int ngay) 65](#_Toc164625992)

[Hình 56. Nội dung của phương thức operator-(int ngay) 67](#_Toc164625993)

[Hình 57. Nội dung của phương thức toDays() const 69](#_Toc164625994)

[Hình 58. Nội dung của phương thức operator-(const NgayThangNam &a) const 70](#_Toc164625995)

[Hình 59. Nội dung của phương thức operator++(int) 71](#_Toc164625996)

[Hình 60. Nội dung của phương thức operator--(int) 73](#_Toc164625997)

[Hình 61. Nội dung của phương thức operator==(NgayThangNam another) 74](#_Toc164625998)

[Hình 62. Nội dung của phương thức operator!=(NgayThangNam another) 75](#_Toc164625999)

[Hình 63. Nội dung của phương thức operator>=(NgayThangNam another) 76](#_Toc164626000)

[Hình 64. Nội dung của phương thức operator<=(NgayThangNam another) 77](#_Toc164626001)

[Hình 65. Nội dung của phương thức operator>(NgayThangNam another) 79](#_Toc164626002)

[Hình 66. Nội dung của phương thức operator<(NgayThangNam another) 80](#_Toc164626003)

[Hình 67. Nội dung của phương thức operator<<(ostream &out, NgayThangNam &ntn) 81](#_Toc164626004)

[Hình 68. Nội dung của phương thức operator>>(ostream &out, NgayThangNam &ntn) 82](#_Toc164626005)

**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:

A black rectangular object with white text

Description automatically generated

Hình 1. Class diagram của lớp PhanSo

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

A screenshot of a computer program

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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.

A white background with black text

Description automatically generated

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.

A screenshot of a computer program

Description automatically generated

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

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

A screenshot of a computer program

Description automatically generated

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

A screenshot of a computer code

Description automatically generated

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;      }      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.

A screenshot of a computer program

Description automatically generated

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;      }      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.

A screenshot of a computer code

Description automatically generated

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.

A computer screen shot of a computer code

Description automatically generated

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.

A white background with black text

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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):

* 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.

A screenshot of a computer code

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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 >>

A screenshot of a computer code

Description automatically generated

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 <<

A computer code with text

Description automatically generated with medium confidence

Hình 17. Nội dung của phương thức operator<<(istream &in, PhanSo &ps)

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

Bảng 21. Nội dung của phương thức operator<<(istream &in, PhanSo &ps)

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:

A black screen with white text

Description automatically generated

Hình 18. Class diagram của lớp SoPhuc

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

A screenshot of a computer program

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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.

A white background with black text

Description automatically generated

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.

A computer code with text

Description automatically generated

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.

A computer screen shot of a computer code

Description automatically generated

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 << "Hieu cua 2 so phuc la: " << 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



A computer code with text

Description automatically generated

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 << "Tich cua 2 so phuc la: " << 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

A mathematical equation with black text

Description automatically generated

A computer code with text

Description automatically generated

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.

A computer screen shot of a computer code

Description automatically generated

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.

A white background with black text

Description automatically generated

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 >>

A screenshot of a computer code

Description automatically generated

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

|  |
| --- |
| 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 <<

A white background with black text

Description automatically generated

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), ++, --, ==, !=, >=, <=, >, >, <<

**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:

A black background with a black square

Description automatically generated with medium confidence

Hình 30. Class diagram của lớp ThoiGian

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

A screenshot of a computer program

Description automatically generated

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.

A screenshot of a computer

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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)

A close up of numbers

Description automatically generated

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.

A screenshot of a computer code

Description automatically generated

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

A screenshot of a computer program

Description automatically generated

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

A screenshot of a computer program

Description automatically generated

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)**.

A screenshot of a computer program

Description automatically generated

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).**

A screenshot of a computer program

Description automatically generated

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)          {              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.

A screenshot of a computer

Description automatically generated

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)

|  |
| --- |
| 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.

A screenshot of a computer

Description automatically generated

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 |
| Output: 22:59:58 |

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**.

A close up of a computer screen

Description automatically generated

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**.

A white background with black and blue text

Description automatically generated

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):

* 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.

A white background with black text

Description automatically generated

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.

A white background with black text

Description automatically generated

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):

* 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.

A white background with black text

Description automatically generated

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.

A white background with black text

Description automatically generated

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):

* Nội dung: Overload toán tử nhập >>

A screenshot of a computer code

Description automatically generated

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

A white background with black text

Description automatically generated

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:

A black background with a black square

Description automatically generated with medium confidence

Hình 50. Class diagram của lớp NgayThangNam

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

A screenshot of a computer program

Description automatically generated

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

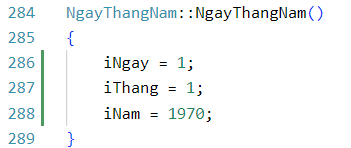
|  |
| --- |
| #include <iostream>  #include <algorithm>  using namespace std;  class NgayThangNam  {  private:      int iNgay, iThang, iNam;  public:      NgayThangNam();      NgayThangNam(int Nam, int Thang, int Ngay = 1);      void TinhNgay();      NgayThangNam operator+(int ngay);      NgayThangNam 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.



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.

A computer screen shot of a computer code

Description automatically generated

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.

A screenshot of a computer program

Description automatically generated

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 << "Ngay thu: " << dayOfYear << " trong nam" << endl;  } |

Bảng 67. Nội dung của phương thức TinhNgay()

|  |
| --- |
| Input: 10/4/2024 |
| Output: Ngay thu: 101 trong nam |

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

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.

A screenshot of a computer program

Description automatically generated

Hình 55. 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 (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):

* 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 => Thang --

3. Nếu số Tháng bé hơn 1 => Năm --, Tháng reset lại = 12.

4. Trả về ngày tháng năm sau khi trừ.

A screenshot of a computer program

Description automatically generated

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.iNgay < 1)      {          temp.iNgay += 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:

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.

A screenshot of a computer program

Description automatically generated

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.

A close-up of a computer screen

Description automatically generated

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

|  |
| --- |
| 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.

A screenshot of a computer program

Description automatically generated

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

|  |
| --- |
| NgayThangNam NgayThangNam::operator++(int)  {      iNgay++;      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.

A screenshot of a computer program

Description automatically generated

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

* Nội dung: So sánh bằng giữa 2 thời điểm

A close up of a computer screen

Description automatically generated

Hình 61. Nội dung của phương thức operator==(NgayThangNam 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

A white background with black text

Description automatically generated

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**.

A screenshot of a computer program

Description automatically generated

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**.

A screenshot of a computer code

Description automatically generated

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<=(NgayThangNam another)

4.1.14 - Phương thức operator>(NgayThangNam another):

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

A screenshot of a computer code

Description automatically generated

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>(NgayThangNam another)

4.1.15 - Phương thức operator<(NgayThangNam another):

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

A screenshot of a computer code

Description automatically generated

Hình 66. 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 86. Nội dung của phương thức operator<(NgayThangNam another)

4.1.16 - Phương thức operator<<(ostream &out, NgayThangNam &ntn):

* Nội dung: Overload toán tử xuất <<.

A white background with black text

Description automatically generated

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

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

Bảng 87. Nội dung của phương thức operator<<(ostream &out, NgayThangNam &ntn)

4.1.17 - Phương thức operator>>(ostream &out, NgayThangNam &ntn):

* Nội dung: Overload toán tử nhập >>.

A screenshot of a computer code

Description automatically generated

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>