Mục Lục

Nội dung

[I. Lab01a 5](#_Toc188289754)

[1. VD1 5](#_Toc188289755)

[Main Idea 5](#_Toc188289756)

[Example 5](#_Toc188289757)

[Source Code 5](#_Toc188289758)

[2. VD2 6](#_Toc188289759)

[Main Idea 6](#_Toc188289760)

[Example 6](#_Toc188289761)

[Source Code 6](#_Toc188289762)

[3. VD3 8](#_Toc188289763)

[Main Idea 8](#_Toc188289764)

[Example 8](#_Toc188289765)

[Source Code 8](#_Toc188289766)

[Mô hình 9](#_Toc188289767)

[4. VD4 10](#_Toc188289768)

[Main Idea 10](#_Toc188289769)

[Example – Result 10](#_Toc188289770)

[Source Code 10](#_Toc188289771)

[5. VD5 12](#_Toc188289772)

[Main Idea 12](#_Toc188289773)

[Example – Result 12](#_Toc188289774)

[Source Code 12](#_Toc188289775)

[II. Lab01b 15](#_Toc188289776)

[1. Bài 1 15](#_Toc188289777)

[Main Idea 15](#_Toc188289778)

[Example – Result 15](#_Toc188289779)

[Source Code 15](#_Toc188289780)

[2. Bài 2 16](#_Toc188289781)

[Main Idea 16](#_Toc188289782)

[Example – Result 16](#_Toc188289783)

[Source Code 16](#_Toc188289784)

[3. Bài 3 17](#_Toc188289785)

[Main Idea 17](#_Toc188289786)

[Example – Result 17](#_Toc188289787)

[Source Code 17](#_Toc188289788)

[4. Bài 4 19](#_Toc188289789)

[Main Idea 19](#_Toc188289790)

[Example – Result 19](#_Toc188289791)

[Source Code 20](#_Toc188289792)

[5. Bài 5 22](#_Toc188289793)

[Main Idea 22](#_Toc188289794)

[Example – Result 22](#_Toc188289795)

[Source Code 22](#_Toc188289796)

[BT phải làm 24](#_Toc188289797)

[1.Bài 1 24](#_Toc188289798)

[2. Bài 2 27](#_Toc188289799)

[3. Bài 3 29](#_Toc188289800)

[4. Bài 4 31](#_Toc188289801)

[5. Bài 5 34](#_Toc188289802)

[III. Lab01c 37](#_Toc188289803)

[1. Bài 1 37](#_Toc188289804)

[Main Idea 37](#_Toc188289805)

[Example – Result 38](#_Toc188289806)

[Source Code 38](#_Toc188289807)

[2. Bài 2 41](#_Toc188289808)

[Main Idea 41](#_Toc188289809)

[Example – Result 41](#_Toc188289810)

[Source Code 42](#_Toc188289811)

[3. Bài 3 45](#_Toc188289812)

[Main Idea 45](#_Toc188289813)

[Example – Result 46](#_Toc188289814)

[Source Code 47](#_Toc188289815)

[BT phải làm 49](#_Toc188289816)

[1.Bài 1 49](#_Toc188289817)

[IV. Lab01d 59](#_Toc188289818)

[1. Bài 1 59](#_Toc188289819)

[Main Idea 59](#_Toc188289820)

[Example -Result 59](#_Toc188289821)

[Source Code 60](#_Toc188289822)

[2. Bài 2 60](#_Toc188289823)

[Main Idea 60](#_Toc188289824)

[Example – Result 61](#_Toc188289825)

[Source Code 62](#_Toc188289826)

[3. Bài 3 63](#_Toc188289827)

[Main Idea 63](#_Toc188289828)

[Example – Result 64](#_Toc188289829)

[Source Code 64](#_Toc188289830)

[BT Phải làm 65](#_Toc188289831)

[1. Bài 1 65](#_Toc188289832)

[2. Bài 2 67](#_Toc188289833)

[3. Bài 3 68](#_Toc188289834)

[V. Lab01e 71](#_Toc188289835)

[1. Bài 1 71](#_Toc188289836)

[Source Code 71](#_Toc188289837)

[2. Bài 2 71](#_Toc188289838)

[Source Code 71](#_Toc188289839)

[BT Phải làm 72](#_Toc188289840)

[1. Bài 1 72](#_Toc188289841)

[2. Bài 2 74](#_Toc188289842)

[3. Bài 3 75](#_Toc188289843)

[Example – Result 77](#_Toc188289844)

[4. Bài 4 78](#_Toc188289845)

[VI. Lab01f 81](#_Toc188289846)

[1. Bài 1 81](#_Toc188289847)

[Main Idea 81](#_Toc188289848)

[Example – Result 81](#_Toc188289849)

[2. Bài 2 82](#_Toc188289850)

[Main Idea 82](#_Toc188289851)

[Example – Result 82](#_Toc188289852)

[Source Code 83](#_Toc188289853)

# I. Lab01a

## 1. VD1

### Main Idea

1. **Biến tĩnh (n) trên stack**: Con trỏ p trỏ đến địa chỉ của n, truy xuất và in giá trị.
2. **Vùng nhớ động**: Con trỏ pn cấp phát động trên heap, gán giá trị, in thông tin, và giải phóng bộ nhớ để tránh rò rỉ.

### Example

Addr = 0x5ffecc, Val = 20

Addr = 0x5ffec0, Val = 0x5ffecc, ValRef = 20

sizeof(int): 4 bytes(s)

Addr = 0x5ffeb8, Val = 0x6ec280, valRef = 10

### Source Code

#include <iostream>

int main()

{

    int n;

    int \*p = nullptr;

    n = 20;

    p = &n;

    std::cout << "Addr = " << &n << ", Val = " << n << "\n";

    std::cout << "Addr = " << &p << ", Val = " << p << ", ValRef = " << \*p << "\n";

    int \*pn = nullptr;

    pn = new int;

    if (pn != nullptr)

    {

        \*pn = 10;

        std::cout << "sizeof(int): " << sizeof(int) << " bytes(s)" << "\n";

        std::cout << "Addr = " << &pn << ", Val = " << pn << ", valRef = " << \*pn << "\n";

        delete[] pn;

        pn = nullptr;

    }

    return 0;

}

## 2. VD2

### Main Idea

1. **Hoán vị bằng truyền tham trị (HoanVi1)**: Hàm nhận hai tham số nguyên và hoán đổi giá trị của chúng. Tuy nhiên, do truyền tham trị, các thay đổi chỉ ảnh hưởng trong phạm vi hàm và không tác động đến biến gốc.
2. **Hoán vị bằng truyền tham chiếu (HoanVi2)**: Hàm nhận hai tham chiếu đến biến nguyên và hoán đổi giá trị của chúng. Vì truyền tham chiếu, các thay đổi ảnh hưởng trực tiếp đến biến gốc.
3. **Hoán vị bằng truyền con trỏ (HoanVi3)**: Hàm nhận hai con trỏ trỏ đến biến nguyên và hoán đổi giá trị tại địa chỉ mà chúng trỏ tới. Cách này cũng thay đổi trực tiếp giá trị của biến gốc
4. **Tham chiếu q đến biến p**: Cả p và q đều tham chiếu đến cùng một vùng nhớ, do đó địa chỉ của chúng giống nhau và thay đổi giá trị qua q sẽ ảnh hưởng đến p.

### Example

x = 5 y = 10

x = 5 y = 10

a = 5 b = 10

a = 10 b = 5

m = 5 n = 10

m = 10 n = 5

Addr = 0x5ffeac, Val = 5

Addr = 0x5ffeac, Val = 5

### Source Code

#include <iostream>

void HoanVi1(int a, int b)

{

    int temp = a;

    a = b;

    b = temp;

}

void HoanVi2(int &a, int &b)

{

    int temp = a;

    a = b;

    b = temp;

}

void HoanVi3(int \*a, int \*b)

{

    int temp = \*a;

    \*a = \*b;

    \*b = temp;

}

int main()

{

    int x = 5, y = 10;

    std::cout << "x = " << x << " y = " << y << "\n";

    HoanVi1(x, y);

    std::cout << "x = " << x << " y = " << y << "\n";

    int a = 5, b = 10;

    std::cout << "a = " << a << " b = " << b << "\n";

    HoanVi2(a, b);

    std::cout << "a = " << a << " b = " << b << "\n";

    int m = 5, n = 10;

    std::cout << "m = " << m << " n = " << n << "\n";

    HoanVi3(&m, &n);

    std::cout << "m = " << m << " n = " << n << "\n";

    int p = 5;

    int &q = p;

    std::cout << "Addr = " << &p << ", Val = " << p << "\n";

    std::cout << "Addr = " << &q << ", Val = " << q << "\n";

    return 0;

}

## 3. VD3

### Main Idea

* 1. **Mảng tĩnh (a) trên stack**: Con trỏ p trỏ vào các phần tử của mảng tĩnh và truy xuất giá trị thông qua chỉ số (vd. p[-2]).
  2. **Mảng động (b) trên heap**: Cấp phát mảng động bằng new int[nb], gán giá trị cho từng phần tử, và giải phóng bộ nhớ bằng delete[] b để tránh rò rỉ.

### Example

Addr = 0x5ffe98, Val = 0x5ffea4, ValRef = 4

### Source Code

#include <iostream>

int main()

{

    int a[5] = {2, 4, 5, 8};

    int na = 3;

    int \*p = nullptr;

    p = a + 1;

    std::cout << "Addr = " << &p << ", Val = " << p << ", ValRef = " << \*p << "\n";

    p = p + 1;

    int x = p[-2];

    int \*b = nullptr;

    int nb = 0;

    nb = 3;

    b = new int[nb];

    \*(b + 0) = 10;

    \*(b + 1) = 20;

    b[2] = 30;

    if (b != nullptr)

    {

        delete[] b;

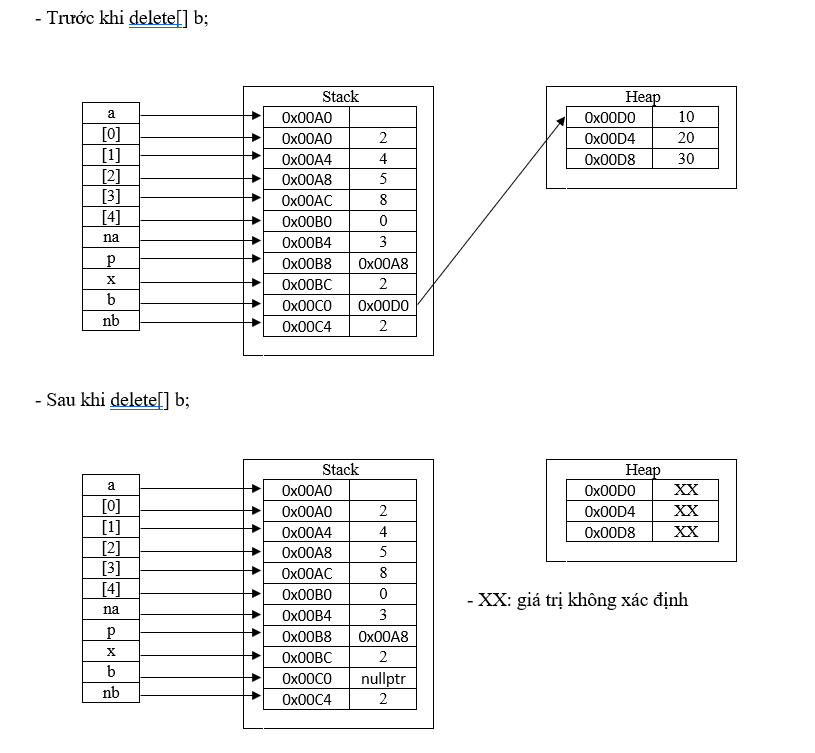
    }

    return 0;

}

### Mô hình

|  |  |  |
| --- | --- | --- |
|  |  |  |



- Sau khi thực hiện delete[] b; : vùng nhớ trên Heap mà ptr b trỏ tới (từ 0x00D0 đến 0x00D8) được giải phóng, giá trị tại các ô nhớ này trở thành không xác định (XX).

- Sau khi delete[] b; và trước return 0; : thì biến b vẫn còn trên Stack nhưng lúc này mang giá trị không hợp lệ nên để tránh lỗi pointer bị treo thì nên gán giá trị nullptr cho b.

## 4. VD4

### Main Idea

1. Nhập mảng
2. Duyệt qua mảng, phần tử nào là số chẵn thì cập nhật lại giá trị phần tử đó của mảng
3. Xuất mảng

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 1 2 3 4 5 | 1 0 3 0 5 |
| 10  1 0 3 8 2 6 5 7 9 4 | 1 0 3 0 0 0 5 7 9 0 |

### Source Code

#include <iostream>

#define MAX 100

void NhapMang(int \*&a, int &n)

{

    std::cout << "Moi ban nhap so luong phan tu: ";

    std::cin >> n;

    a = new int[n];

    for (int i = 0; i <= n - 1; i++)

    {

        std::cout << "Phan tu " << i << ": ";

        std::cin >> a[i];

    }

}

void NhapMang(int \*\*a, int \*n)

{

    std::cout << "Moi ban nhap so luong phan tu: ";

    std::cin >> \*n;

    \*a = new int[\*n];

    for (int i = 0; i <= \*n - 1; i++)

    {

        std::cout << "Phan tu " << i << ": ";

        std::cin >> (\*a)[i];

    }

}

void XuatMang(int \*a, int n)

{

    std::cout << "Day so co " << n << " phan tu: ";

    for (int i = 0; i <= n - 1; i++)

    {

        std::cout << a[i] << " ";

    }

    std::cout << std::endl;

}

void DoiChan(int \*a, int n)

{

    for (int i = 0; i <= n - 1; i++)

    {

        if (a[i] % 2 == 0)

        {

            a[i] = 0;

        }

    }

}

int main()

{

    int \*b = nullptr;

    int k = 0;

    NhapMang(&b, &k);

    XuatMang(b, k);

    DoiChan(b, k);

    std::cout << "\* Doi gia tri chan thanh cac so 0: \n";

    XuatMang(b, k);

    if (b != nullptr)

        delete[] b;

    return 0;

}

## 5. VD5

### Main Idea

1. **Nhập mảng**: Người dùng nhập số lượng và giá trị của mảng động.
2. **Tách số chẵn**: Đếm và sao chép các số chẵn từ mảng gốc sang mảng mới.
3. **Xuất mảng số chẵn**: In ra mảng chứa các số chẵn.
4. **Giải phóng bộ nhớ**: Xóa vùng nhớ cấp phát động sau khi sử dụng.

- Ở đây dùng con trỏ cấp 2 (\*\*a) có thể thay thế bằng tham chiếu (\*&a)

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 1 2 3 4 5 | 2 4 |
| 10  1 1 3 3 5 5 7 7 9 9 | “ ” |

### Source Code

#include <iostream>

#define MAX 100

void NhapMang(int \*\*a, int \*n)

{

    std::cout << "Moi ban nhap so luong phan tu: ";

    std::cin >> \*n;

    while (\*n <= 0 || \*n > MAX)

    {

        std::cout << "Nhap sai! Nhap lai: ";

        std::cin >> \*n;

    }

    \*a = new int[\*n];

    for (int i = 0; i < \*n; i++)

    {

        std::cout << "Phan tu " << i << ": ";

        std::cin >> (\*a)[i];

    }

}

void XuatMang(int \*a, int n)

{

    for (int i = 0; i < n; i++)

    {

        std::cout << a[i] << " ";

    }

    std::cout << std::endl;

}

int DemChan(int \*a, int n)

{

    int dem = 0;

    for (int i = 0; i < n; i++)

    {

        if (a[i] % 2 == 0)

            dem++;

    }

    return dem;

}

void TachChan(int \*a, int n, int \*\*b, int \*m)

{

    \*m = DemChan(a, n);

    \*b = new int[\*m];

    int cs = 0;

    for (int i = 0; i < n; i++)

    {

        if (a[i] % 2 == 0)

        {

            (\*b)[cs++] = a[i];

        }

    }

}

int main()

{

    int \*b = nullptr;

    int k = 0;

    int \*aChan = nullptr;

    int nChan = 0;

    NhapMang(&b, &k);

    TachChan(b, k, &aChan, &nChan);

    if (nChan > 0)

    {

        std::cout << "+ Day chua cac so chan\n";

        XuatMang(aChan, nChan);

    }

    else

    {

        std::cout << "Khong co day phu hop\n";

    }

    delete[] b;

    delete[] aChan;

    return 0;

}

# II. Lab01b

## 1. Bài 1

### Main Idea

- Xây dựng hàm Swap với 2 tham chiếu là pointer

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 10 | 10 5 |
| 1 1 | 1 1 |

### Source Code

#include <iostream>

void Swap(int \*a, int \*b)

{

    int temp = \*a;

    \*a = \*b;

    \*b = temp;

}

int main()

{

    int x, y;

    std::cout << "Moi ban nhap so a: ";

    std::cin >> x;

    std::cout << "Moi ban nhap so b: ";

    std::cin >> y;

    std::cout << "Truoc khi hoan vi, a = " << x << " va b = " << y << ".\n";

    Swap(&x, &y);

    std::cout << "Sau khi hoan vi, a = " << x << " va b = " << y << ".\n";

    return 0;

}

## 2. Bài 2

### Main Idea

- Số chính phương là bình phương của một số, để kiểm tra t kiểm tra thử căn bậc 2 của số đó khi bình phương lại có bằng chính nó hay không.

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 | sai |
| 25 | dung |

### Source Code

#include <iostream>

#include <cmath>

bool checkSquare(int \*n)

{

    int temp = sqrt(\*n);

    return temp \* temp == \*n;

}

int main()

{

    int x;

    int \*ptr = &x;

    std::cout << "Moi ban nhap so nguyen n: ";

    std::cin >> x;

    if (checkSquare(ptr))

    {

        std::cout << x << " la so chinh phuong.\n";

    }

    else

    {

        std::cout << x << " khong phai la so chinh phuong.\n";

    }

    return 0;

}

## 3. Bài 3

### Main Idea

1. **Nhập mảng**: Người dùng nhập số lượng và giá trị của mảng động.
2. **Tách số chẵn**: Đếm và sao chép các số chẵn từ mảng gốc sang mảng mới.
3. **Xuất mảng số chẵn**: In ra mảng chứa các số chẵn.
4. **Giải phóng bộ nhớ**: Xóa vùng nhớ cấp phát động sau khi sử dụng.

- Ở đây dùng con trỏ cấp 2 (\*\*a) có thể thay thế bằng tham chiếu (\*&a)

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 1 2 3 4 5 | 2 4 |
| 10  1 1 3 3 5 5 7 7 9 9 | “ ” |

### Source Code

#include <iostream>

#define MAX 100

void NhapMang(int \*\*a, int \*n)

{

    std::cout << "Moi ban nhap so luong phan tu: ";

    std::cin >> \*n;

    while (\*n <= 0 || \*n > MAX)

    {

        std::cout << "Nhap sai! Nhap lai: ";

        std::cin >> \*n;

    }

    \*a = new int[\*n];

    for (int i = 0; i < \*n; i++)

    {

        std::cout << "Phan tu " << i << ": ";

        std::cin >> (\*a)[i];

    }

}

void XuatMang(int \*a, int n)

{

    for (int i = 0; i < n; i++)

    {

        std::cout << a[i] << " ";

    }

    std::cout << std::endl;

}

int DemChan(int \*a, int n)

{

    int dem = 0;

    for (int i = 0; i < n; i++)

    {

        if (a[i] % 2 == 0)

            dem++;

    }

    return dem;

}

void TachChan(int \*a, int n, int \*\*b, int \*m)

{

    \*m = DemChan(a, n);

    \*b = new int[\*m];

    int cs = 0;

    for (int i = 0; i < n; i++)

    {

        if (a[i] % 2 == 0)

        {

            (\*b)[cs++] = a[i];

        }

    }

}

int main()

{

    int \*b = nullptr;

    int k = 0;

    int \*aChan = nullptr;

    int nChan = 0;

    NhapMang(&b, &k);

    TachChan(b, k, &aChan, &nChan);

    if (nChan > 0)

    {

        std::cout << "+ Day chua cac so chan\n";

        XuatMang(aChan, nChan);

    }

    else

    {

        std::cout << "Khong co day phu hop\n";

    }

    delete[] b;

    delete[] aChan;

    return 0;

}

## 4. Bài 4

### Main Idea

- Kiếm tra tính đan xen: kiểm tra 2 số liền kề bằng tích của chúng, nếu nhỏ hơn 0 thì tiếp tục xét, nếu lớn hơn 0 thì trả về false.

- Kiểm tra tính đồng điệu (tăng dần): nếu số sau nhỏ hơn số trước thì trả về false.

- Kiểm tra tính đối xứng: xét số đầu và số cuối, nếu 2 số giống nhau thì tiếp tục xét các số sau và trước nó.

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 -2 5 -6 5 -2 | Day co tinh chat dan dau.  Day khong co tinh chat don dieu.  Day co tinh chat doi xung. |
| 5  1 2 3 4 5 | Day khong co tinh chat dan dau.  Day co tinh chat don dieu.  Day khong co tinh chat doi xung. |

### Source Code

#include <iostream>

bool checkAlternating(int \*a, int n)

{

    for (int i = 0; i < n - 1; i++)

    {

        if (a[i] \* a[i + 1] >= 0)

            return false;

    }

    return true;

}

bool checkMonoIncreasing(int \*a, int n)

{

    for (int i = 0; i < n - 1; i++)

    {

        if (a[i] > a[i + 1])

        {

            return false;

        }

    }

    return true;

}

bool checkSymmetry(int \*a, int n)

{

    for (int i = 0; i < n / 2; i++)

    {

        if (a[i] != a[n - i - 1])

        {

            return false;

        }

    }

    return true;

}

int main()

{

    int n;

    std::cin >> n;

    int \*a = new int[n];

    for (int i = 0; i < n; i++)

    {

        std::cin >> a[i];

    }

    if (checkAlternating(a, n))

    {

        std::cout << "Day co tinh chat dan dau.\n";

    }

    else

    {

        std::cout << "Day khong co tinh chat dan dau.\n";

    }

    if (checkMonoIncreasing(a, n))

    {

        std::cout << "Day co tinh don dieu.\n";

    }

    else

    {

        std::cout << "Day khong co tinh don dieu.\n";

    }

    if (checkSymmetry(a, n))

    {

        std::cout << "Day co tinh chat doi xung.\n";

    }

    else

    {

        std::cout << "Day khong co tinh chat doi xung.\n";

    }

    delete[] a;

    return 0;

}

## 5. Bài 5

### Main Idea

1. **Nhập mảng**: Người dùng nhập số lượng và giá trị của mảng động.
2. **Tách số chẵn**: Đếm và sao chép các số chẵn từ mảng gốc sang mảng mới.
3. **Xuất mảng số chẵn**: In ra mảng chứa các số chẵn.
4. **Giải phóng bộ nhớ**: Xóa vùng nhớ cấp phát động sau khi sử dụng.

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5 1 2 3 4 5 | 2 3 5 |
| 10  0 2 3 7 8 9 10 15 19 22 | 2 3 7 19 |

### Source Code

#include <iostream>

#include <cmath>

#define MAX 100

void input(int \*\*a, int \*n)

{

    std::cout << "Moi ban nhap so luong phan tu: ";

    std::cin >> \*n;

    while (\*n <= 0 || \*n > MAX)

    {

        std::cout << "Nhap sai! Nhap lai: ";

        std::cin >> \*n;

    }

    \*a = new int[\*n];

    for (int i = 0; i < \*n; i++)

    {

        std::cout << "Phan tu " << i << ": ";

        std::cin >> (\*a)[i];

    }

}

void output(int \*a, int n)

{

    for (int i = 0; i < n; i++)

    {

        std::cout << a[i] << " ";

    }

    std::cout << std::endl;

}

bool checkPrime(int n)

{

    if (n < 2)

        return false;

    int k = sqrt(n);

    for (int i = 2; i <= k; i++)

    {

        if (n % i == 0)

            return false;

    }

    return true;

}

void sepePrime(int \*a, int n, int \*\*b, int \*m)

{

    \*m = 0;

    for (int i = 0; i < n; i++)

    {

        if (checkPrime(a[i]))

        {

            (\*m)++;

        }

    }

    if (\*m > 0)

    {

        \*b = new int[\*m];

        int cs = 0;

        for (int i = 0; i < n; i++)

        {

            if (checkPrime(a[i]))

            {

                (\*b)[cs++] = a[i];

            }

        }

    }

}

int main()

{

    int \*b = nullptr;

    int k = 0;

    int \*aPrime = nullptr;

    int nPrime = 0;

    input(&b, &k);

    sepePrime(b, k, &aPrime, &nPrime);

    if (nPrime > 0)

    {

        std::cout << "Day chua cac so nguyen to\n";

        output(aPrime, nPrime);

    }

    else

    {

        std::cout << "Khong co day phu hop\n";

    }

    delete[] b;

    delete[] aPrime;

    return 0;

}

## BT phải làm

### 1.Bài 1

#### Main Idea

1. Xét hệ số a để xác định loại phương trình (bậc nhất hay bậc hai):
2. Nếu a = 0: Kiểm tra tiếp b và c để xác định có nghiệm, vô nghiệm, hay vô số nghiệm.
3. Nếu a ≠ 0: Tính delta (d = b^2 - 4ac) để xác định số nghiệm:
   1. d < 0: Phương trình vô nghiệm.
   2. d = 0: Phương trình có nghiệm kép.
   3. d > 0: Phương trình có hai nghiệm phân biệt.
4. Trực tiếp tính và in kết quả, đảm bảo hiển thị đủ chính xác dựa vào giá trị của phép tính.

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 0 0 0 | vo so nghiem |
| 0 0 5 | vo nghiem |
| 0 2 -4 | nghiem duy nhat = 2 |
| 1 2 5 | vo nghiem |
| 1 -2 1 | nghiem kep = 1 |
| 1 -3 2 | 2 nghiem pb 1 va 2 |
| 1 -1 -2 | 2 nghiem pb -1 va 2 |

#### Source Code

#include <iostream>

#include <cmath>

void solve(int a, int b, int c, double \*x1, double \*x2, int \*numSolutions)

{

    if (a == 0)

    {

        if (b == 0)

        {

            if (c == 0)

            {

                \*numSolutions = -1; // Vô số nghiệm

            }

            else

            {

                \*numSolutions = 0; // Vô nghiệm

            }

        }

        else

        {

            \*numSolutions = 1; // Nghiệm duy nhất

            \*x1 = -1.0 \* c / b;

        }

    }

    else

    {

        double delta = b \* b - 4 \* a \* c;

        if (delta < 0)

        {

            \*numSolutions = 0; // Vô nghiệm

        }

        else if (delta == 0)

        {

            \*numSolutions = 1; // Nghiệm kép

            \*x1 = -1.0 \* b / (2 \* a);

        }

        else

        {

            \*numSolutions = 2; // Hai nghiệm phân biệt

            \*x1 = (-b + sqrt(delta)) / (2 \* a);

            \*x2 = (-b - sqrt(delta)) / (2 \* a);

            // Đảm bảo nghiệm nhỏ hơn là x1, lớn hơn là x2

            if (\*x1 > \*x2)

            {

                std::swap(\*x1, \*x2);

            }

        }

    }

}

int main()

{

    int a, b, c;

    double x1, x2;

    int numSolutions;

    std::cout << "Enter the values of a, b, c: ";

    std::cin >> a >> b >> c;

    solve(a, b, c, &x1, &x2, &numSolutions);

    // In kết quả

    if (numSolutions == -1)

    {

        std::cout << "Vo so nghiem\n";

    }

    else if (numSolutions == 0)

    {

        std::cout << "Vo nghiem\n";

    }

    else if (numSolutions == 1)

    {

        std::cout << "Nghiem duy nhat: " << x1 << "\n";

    }

    else if (numSolutions == 2)

    {

        std::cout << "2 nghiem phan biet: " << x1 << " and " << x2 << "\n";

    }

    return 0;

}

### 2. Bài 2

#### Main Idea

1. **Hàm checkEvenAndOdd**:

* Duyệt qua từng cặp số liên tiếp trong mảng.
* Nếu hai số liên tiếp cùng là số chẵn hoặc cùng là số lẻ, trả về false.
* Nếu không có cặp nào vi phạm, trả về true.

1. **Hàm checkEven**:

* Duyệt qua từng phần tử trong mảng.
* Nếu gặp số lẻ, trả về false.
* Nếu tất cả đều là số chẵn, trả về true.

|  |  |
| --- | --- |
| Input | Output |
| 5 -2 5 -6 5 -2 | Day co tinh chat chan le  Day khong co tinh chat chan |
| 5  2 2 2 2 2 | Day khong co tinh chat chan le  Day co tinh chat chan |

Examle – Result

#### Source Code

#include <iostream>

bool checkEvenAndOdd(int \*a, int n)

{

    for (int i = 0; i < n - 1; i++)

    {

        if (a[i] % 2 == a[i + 1] % 2)

            return false;

    }

    return true;

}

bool checkEven(int \*a, int n)

{

    for (int i = 0; i < n - 1; i++)

    {

        if (a[i] % 2 != 0)

        {

            return false;

        }

    }

    return true;

}

int main()

{

    int n;

    std::cin >> n;

    int \*a = new int[n];

    for (int i = 0; i < n; i++)

    {

        std::cin >> a[i];

    }

    if (checkEvenAndOdd(a, n))

    {

        std::cout << "Day co tinh chat chan le.\n";

    }

    else

    {

        std::cout << "Day khong co tinh chat chan le.\n";

    }

    if (checkEven(a, n))

    {

        std::cout << "Day co tinh chat chan.\n";

    }

    else

    {

        std::cout << "Day khong co tinh chat chan.\n";

    }

    delete[] a;

    return 0;

}

### 3. Bài 3

#### Main Idea

1. **Kiểm tra nguyên âm:**

* Chúng ta sử dụng một hàm checkVowel để kiểm tra xem một ký tự có phải là nguyên âm (a, e, i, o, u) không. Hàm này chuyển ký tự về dạng chữ thường trước khi so sánh.

1. **Tách nguyên âm:**

* Hàm sepeVowels sẽ:

|  |  |
| --- | --- |
| Input | Output |
| 5 a B c E i | a e i |
| 6  Z y u O t E | u o e |

* Đếm số lượng nguyên âm trong mảng nhập vào.
* Cấp phát bộ nhớ cho một mảng con b chứa tất cả các nguyên âm từ mảng nhập vào.
* Lưu các nguyên âm vào mảng b.

#### Example – Result

#### Source Code

#include <iostream>

bool checkVowel(char c)

{

    c = tolower(c);

    if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')

    {

        return true;

    }

    return false;

}

void sepeVowels(char \*a, int n, char \*\*b, int \*count)

{

    \*count = 0;

    for (int i = 0; i < n; i++)

    {

        if (checkVowel(a[i]))

        {

            (\*count)++;

        }

    }

    \*b = new char[\*count];

    int index = 0;

    for (int i = 0; i < n; i++)

    {

        if (checkVowel(a[i]))

        {

            (\*b)[index++] = a[i];

        }

    }

}

int main()

{

    int n;

    std::cin >> n;

    char \*a = new char[n];

    for (int i = 0; i < n; i++)

    {

        std::cin >> a[i];

    }

    char \*b = nullptr;

    int count = 0;

    sepeVowels(a, n, &b, &count);

    for (int i = 0; i < count; i++)

    {

        std::cout << b[i] << " ";

    }

    std::cout << std::endl;

    delete[] a;

    delete[] b;

    return 0;

}

### 4. Bài 4

#### Main Idea

Hàm hợp nhất hai mảng đã sắp xếp:

Nhập mảng a với kích thước sizeA

Nhập mảng b với kích thước sizeB

Tạo mảng kết quả c với kích thước sizeA + sizeB

Khởi tạo các chỉ số index1 = 0, index2 = 0, index3 = 0

Trong khi index1 < sizeA và index2 < sizeB:

Nếu a[index1] < b[index2]:

c[index3] = a[index1]

index1 = index1 + 1

Ngược lại:

c[index3] = b[index2]

index2 = index2 + 1

index3 = index3 + 1

Nếu còn phần tử trong mảng a:

Lặp qua mảng a từ index1 đến sizeA, thêm phần tử vào c

Nếu còn phần tử trong mảng b:

Lặp qua mảng b từ index2 đến sizeB, thêm phần tử vào c

In ra mảng c

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 2  2 5  3  1 3 6 | 1 2 3 5 6 |
| 0  3  2 5 6 | 2 5 6 |

#### Source Code

#include <iostream>

void stringConcatenation(double \*a, int sizeA, double \*b, int sizeB, double \*\*c, int \*sizeC)

{

    \*sizeC = sizeA + sizeB;

    \*c = new double[\*sizeC];

    int index1 = 0;

    int index2 = 0;

    int index3 = 0;

    while (index1 < sizeA && index2 < sizeB)

    {

        if (a[index1] < b[index2])

        {

            (\*c)[index3++] = a[index1++];

        }

        else

        {

            (\*c)[index3++] = b[index2++];

        }

    }

    while (index1 < sizeA)

    {

        (\*c)[index3++] = a[index1++];

    }

    while (index2 < sizeB)

    {

        (\*c)[index3++] = b[index2++];

    }

}

int main()

{

    int m;

    std::cin >> m;

    double \*a = new double[m];

    for (int i = 0; i < m; i++)

    {

        std::cin >> a[i];

    }

    int n;

    std::cin >> n;

    double \*b = new double[n];

    for (int i = 0; i < n; i++)

    {

        std::cin >> b[i];

    }

    double \*c = nullptr;

    int size = 0;

    stringConcatenation(a, m, b, n, &c, &size);

    for (int i = 0; i < size; i++)

    {

        std::cout << c[i] << " ";

    }

    delete[] a;

    delete[] b;

    delete[] c;

    return 0;

}

### 5. Bài 5

#### Main Idea

Bắt đầu:

1. Nhập số phần tử n.

2. Tạo mảng a với n phần tử.

3. Nhập các phần tử vào mảng a.

4. Tạo mảng b và c, khởi tạo sizeB = 0, sizeC = 0.

5. Duyệt qua mảng a:

- Nếu a[i] là chẵn, thêm vào mảng b và tăng sizeB.

- Nếu a[i] là lẻ, thêm vào mảng c và tăng sizeC.

6. In các phần tử trong mảng b (các số chẵn).

7. In các phần tử trong mảng c (các số lẻ).

8. Giải phóng bộ nhớ đã cấp phát cho mảng a, b, c.

Kết thúc.

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 5  1 2 3 4 5 | 2 4  1 3 5 |
| 6  7 8 12 13 15 20 | 7 13 15  8 12 20 |

#### Source Code

#include <iostream>

void sepeString(int \*a, int sizeA, int \*\*b, int \*sizeB, int \*\*c, int \*sizeC)

{

    \*sizeB = 0;

    \*sizeC = 0;

    \*b = new int[sizeA];

    \*c = new int[sizeA];

    for (int i = 0; i < sizeA; i++)

    {

        if (a[i] % 2 == 0)

        {

            (\*b)[(\*sizeB)++] = a[i];

        }

        else

        {

            (\*c)[(\*sizeC)++] = a[i];

        }

    }

}

int main()

{

    int n;

    std::cin >> n;

    int \*a = new int[n];

    for (int i = 0; i < n; i++)

    {

        std::cin >> a[i];

    }

    int \*b = nullptr;

    int sizeB = 0;

    int \*c = nullptr;

    int sizeC = 0;

    sepeString(a, n, &b, &sizeB, &c, &sizeC);

    for (int i = 0; i < sizeB; i++)

    {

        std::cout << b[i] << " ";

    }

    std::cout << "\n";

    for (int i = 0; i < sizeC; i++)

    {

        std::cout << c[i] << " ";

    }

    delete[] a;

    delete[] b;

    delete[] c;

    return 0;

}

# III. Lab01c

## 1. Bài 1

### Main Idea

STRUCT Fraction

numer // Tử số

denom // Mẫu số

FUNCTION enterFraction(Fraction &frac)

INPUT frac.numer

INPUT frac.denom

FUNCTION printFraction(Fraction frac)

PRINT frac.numer "/" frac.denom

FUNCTION GCD(a, b)

IF a % b == 0

THEN RETURN b

ELSE

RETURN GCD(b, a % b)

FUNCTION reduceFraction(Fraction &frac)

gcd = GCD(frac.numer, frac.denom)

frac.numer = frac.numer / gcd

frac.denom = frac.denom / gcd

FUNCTION compareFraction(Fraction frac1, Fraction frac2)

value = (frac1.numer / frac1.denom) - (frac2.numer / frac2.denom)

IF value > 0

THEN RETURN 1

ELSE IF value == 0

THEN RETURN 0

ELSE

RETURN -1

FUNCTION addFraction(Fraction frac1, Fraction frac2)

FUNCTION subtractFraction(Fraction frac1, Fraction frac2)

FUNCTION multiplyFraction(Fraction frac1, Fraction frac2)

FUNCTION divideFraction(Fraction frac1, Fraction frac2)

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 4 8 2 5 | 4/8  2/5  1/2  2/5  -1  9/10  -1/10  1/5  5/4 |
| 2 4 3 6 | 2/4  3/6  1/2  1/2  0  1/1  0/1  1/4  1/1 |

### Source Code

#include <iostream>

struct Fraction

{

    int numer;

    int denom;

};

void enterFraction(Fraction &frac)

{

    std::cout << "Enter numerator: ";

    std::cin >> frac.numer;

    std::cout << "Enter denominator; ";

    std::cin >> frac.denom;

}

void printFraction(Fraction frac)

{

    std::cout << frac.numer << "/" << frac.denom << "\n";

}

int GCD(int a, int b)

{

    if (a % b == 0)

        return b;

    return GCD(b, a % b);

}

void reduceFraction(Fraction &frac)

{

    int gcd = GCD(frac.numer, frac.denom);

    frac.numer /= gcd;

    frac.denom /= gcd;

}

int compareFraction(Fraction frac1, Fraction frac2)

{

    double value = (double)frac1.numer / frac1.denom - (double)frac2.numer / frac2.denom;

    if (value > 0)

    {

        return 1;

    }

    else if (value == 0)

    {

        return 0;

    }

    else

    {

        return -1;

    }

}

Fraction addFraction(Fraction frac1, Fraction frac2)

{

    Fraction result;

    result.numer = frac1.numer \* frac2.denom + frac2.numer \* frac1.denom;

    result.denom = frac1.denom \* frac2.denom;

    return result;

}

Fraction subtractFraction(Fraction frac1, Fraction frac2)

{

    Fraction result;

    result.numer = frac1.numer \* frac2.denom - frac2.numer \* frac1.denom;

    result.denom = frac1.denom \* frac2.denom;

    return result;

}

Fraction multiplyFraction(Fraction frac1, Fraction frac2)

{

    Fraction result;

    result.numer = frac1.numer \* frac2.numer;

    result.denom = frac1.denom \* frac2.denom;

    return result;

}

Fraction divideFraction(Fraction frac1, Fraction frac2)

{

    Fraction result;

    result.numer = frac1.numer \* frac2.denom;

    result.denom = frac1.denom \* frac2.numer;

    return result;

}

int main() {

    Fraction frac1, frac2;

    enterFraction(frac1);

    enterFraction(frac2);

    printFraction(frac1);

    printFraction(frac2);

    reduceFraction(frac1);

    reduceFraction(frac2);

    printFraction(frac1);

    printFraction(frac2);

    std::cout << compareFraction(frac1, frac2) << "\n";

    printFraction(addFraction(frac1, frac2));

    printFraction(subtractFraction(frac1, frac2));

    printFraction(multiplyFraction(frac1, frac2));

    printFraction(divideFraction(frac1, frac2));

    return 0;

}

## 2. Bài 2

### Main Idea

Hàm inputOneBook(book):

- Nhập tiêu đề, ID, giá của sách

- Cập nhật thông tin vào đối tượng book

Hàm inputBookArray(books[], n):

- Nhập số lượng sách n

- Lặp qua n cuốn sách, nhập thông tin cho mỗi cuốn sách

Hàm searchBookByID(books[], n, id):

- Lặp qua các sách:

- Nếu sách có ID trùng với id tìm kiếm, trả về true

- Nếu không tìm thấy, trả về false

Hàm searchBookByTitle(books[], n, title):

- Đổi tiêu đề sách và tiêu đề tìm kiếm thành chữ thường

- Lặp qua các sách:

- Nếu sách có tiêu đề trùng với title tìm kiếm, trả về true

- Nếu không tìm thấy, trả về false

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| Enter number of books: 3  Enter book 1  Enter title: The Great Gatsby  Enter ID: B001  Enter price: 500  Enter book 2  Enter title: Moby Dick  Enter ID: B002  Enter price: 600  Enter book 3  Enter title: To Kill a Mockingbird  Enter ID: B003  Enter price: 700  Enter ID to search: B002  Enter title to search: Moby Dick | Book found by ID!  Book found by ID! |

### Source Code

#include <iostream>

#include <string>

class Book

{

private:

    std::string title;

    std::string id;

    int price;

public:

    std::string getTitle()

    {

        return title;

    }

    void setTitle(std::string title)

    {

        this->title = title;

    }

    std::string getID()

    {

        return id;

    }

    void setID(std::string id)

    {

        this->id = id;

    }

    int getPrice()

    {

        return price;

    }

    void setPrice(int price)

    {

        this->price = price;

    }

    void displayInfo()

    {

        std::cout << "Title: " << title << "\n";

        std::cout << "ID: " << id << "\n";

        std::cout << "Price: " << price << "\n";

    }

};

void inputOneBook(Book &book)

{

    std::string title, id;

    int price;

    std::cout << "Enter title: ";

    std::cin.ignore(); // Clear buffer

    std::getline(std::cin, title);

    std::cout << "Enter ID: ";

    std::getline(std::cin, id);

    std::cout << "Enter price: ";

    std::cin >> price;

    book.setTitle(title);

    book.setID(id);

    book.setPrice(price);

}

void inputBookArray(Book books[], int &n)

{

    for (int i = 0; i < n; i++)

    {

        std::cout << "Enter book " << i + 1 << "\n";

        inputOneBook(books[i]);

    }

}

bool searchBookByID(Book books[], int n, const std::string &id)

{

    for (int i = 0; i < n; i++)

    {

        if (books[i].getID() == id)

        {

            return true;

        }

    }

    return false;

}

// Search by Book Title

bool searchBookByTitle(Book books[], int n, const std::string &title)

{

    std::string lowerTitle = title;

    for (char &c : lowerTitle)

    {

        c = tolower(c);

    }

    for (int i = 0; i < n; i++)

    {

        std::string bookTitle = books[i].getTitle();

        for (char &c : bookTitle)

        {

            c = tolower(c);

        }

        if (bookTitle == lowerTitle)

        {

            return true;

        }

    }

    return false;

}

int main()

{

    int n;

    std::cout << "Enter number of books: ";

    std::cin >> n;

    Book books[n];

    inputBookArray(books, n);

    std::string searchID;

    std::cout << "Enter ID to search: ";

    std::cin >> searchID;

    if (searchBookByID(books, n, searchID))

    {

        std::cout << "Book found by ID!\n";

    }

    else

    {

        std::cout << "Book not found by ID.\n";

    }

    std::string searchTitle;

    std::cout << "Enter title to search: ";

    std::cin.ignore(); // Clear buffer

    std::getline(std::cin, searchTitle);

    if (searchBookByTitle(books, n, searchTitle))

    {

        std::cout << "Book found by title!\n";

    }

    else

    {

        std::cout << "Book not found by title.\n";

    }

    return 0;

}

## 3. Bài 3

### Main Idea

DEFINE Student STRUCT (id, firstName, lastName, sex, yoB)

DEFINE Class STRUCT (student[50], quantity)

FUNCTION inputStudentInfo(STUDENT):

INPUT id, firstName, lastName, sex, yoB

FUNCTION inputClassStudent(CLASS &classStudent, INTEGER n):

FOR i = 0 TO n-1:

CALL inputStudentInfo(classStudent.student[i])

FUNCTION countStudentBySex(CLASS classStudent, INTEGER &quantityMan, INTEGER &quantityWoman):

FOR i = 0 TO classStudent.quantity-1:

IF sex == 1: quantityMan++;

ELSE: quantityWoman++

FUNCTION sortByYoB(CLASS &classStudent):

FOR i = 0 TO n-2:

FOR j = i+1 TO n-1:

IF student[i].yoB > student[j].yoB:

SWAP student[i] and student[j]

|  |  |
| --- | --- |
| Input | Output |
| Enter the number of students: 3  Enter student 1  Enter ID: 001  Enter First Name: John  Enter Last Name: Doe  Enter Sex (1 for Man, 0 for Woman): 1  Enter Year of Birth: 2000  Enter student 2  Enter ID: 002  Enter First Name: Jane  Enter Last Name: Smith  Enter Sex (1 for Man, 0 for Woman): 0  Enter Year of Birth: 1998  Enter student 3  Enter ID: 003  Enter First Name: Bob  Enter Last Name: Lee  Enter Sex (1 for Man, 0 for Woman): 1  Enter Year of Birth: 2001 | Number of Men: 2  Number of Women: 1  Sorted by Year of Birth:  ID: 002, Name: Jane Smith, Year of Birth: 1998  ID: 001, Name: John Doe, Year of Birth: 2000  ID: 003, Name: Bob Lee, Year of Birth: 2001 |

### Example – Result

### Source Code

#include <iostream>

#include <string>

struct Student

{

    std::string id;

    std::string firstName;

    std::string lastName;

    int sex; // Man = 1 and Woman = 0

    int yoB; // Year of Birth

};

struct Class

{

    Student student[50];

    int quantity;

};

void inputStudentInfo(Student &student)

{

    std::cout << "Enter ID: ";

    std::cin >> student.id;

    std::cout << "Enter First Name: ";

    std::cin >> student.firstName;

    std::cout << "Enter Last Name: ";

    std::cin >> student.lastName;

    std::cout << "Enter Sex: ";

    std::cin >> student.sex;

    std::cout << "Enter Year of Birth: ";

    std::cin >> student.yoB;

}

void inputClassStudent(Class &classStudent, int n)

{

    classStudent.quantity = n;

    for (int i = 0; i < n; i++)

    {

        std::cout << "Enter student " << i + 1 << "\n";

        inputStudentInfo(classStudent.student[i]);

    }

}

void countStudentBySex(Class &classStudent, int &quantityMan, int &quantityWoman)

{

    quantityMan = 0;

    quantityWoman = 0;

    for (int i = 0; i < classStudent.quantity; i++)

    {

        if (classStudent.student[i].sex == 1)

        {

            quantityMan++;

        }

        else

        {

            quantityWoman++;

        }

    }

}

void sortByYoB(Class &classStudent)

{

    for (int i = 0; i < classStudent.quantity - 1; i++)

    {

        for (int j = i + 1; j < classStudent.quantity; j++)

        {

            if (classStudent.student[i].yoB > classStudent.student[j].yoB)

            {

                Student temp = classStudent.student[i];

                classStudent.student[i] = classStudent.student[j];

                classStudent.student[j] = temp;

            }

        }

    }

}

int main()

{

    Class classStudent;

    int n;

    std::cout << "Enter the number of students: ";

    std::cin >> n;

    inputClassStudent(classStudent, n);

    // Count students by sex

    int quantityMan, quantityWoman;

    countStudentBySex(classStudent, quantityMan, quantityWoman);

    std::cout << "Number of Men: " << quantityMan << "\n";

    std::cout << "Number of Women: " << quantityWoman << "\n";

    // Sort by Year of Birth

    sortByYoB(classStudent);

    std::cout << "Sorted by Year of Birth: \n";

    for (int i = 0; i < classStudent.quantity; i++)

    {

        std::cout << "ID: " << classStudent.student[i].id << ", Name: "

                  << classStudent.student[i].firstName << " " << classStudent.student[i].lastName

                  << ", Year of Birth: " << classStudent.student[i].yoB << "\n";

    }

    return 0;

}

## BT phải làm

### 1.Bài 1

#### Main Idea

DEFINE Employee STRUCT (id, firstName, lastName, gender, seniority)

DEFINE Department STRUCT (employee[100], quantity)

FUNCTION inputEmployeeInfo(EMPLOYEE):

INPUT id, firstName, lastName, gender, yoB

FUNCTION inputDepartmentEmployee(DEPARTMENT &departmentEmployee, INTEGER n):

FOR i = 0 TO n-1:

CALL inputEmployeeInfo(departmentEmployee.Employee[i])

FUNCTION countEmployeeByGender(DEPARTMENT departmentEmployee, INTEGER &quantityMan, INTEGER &quantityWoman):

FOR i = 0 TO departmentEmployee.quantity-1:

IF sex == 1: quantityMan++;

ELSE: quantityWoman++

FUNCTION sortBySeniority(DEPARTMENT &classStudent):

FOR i = 0 TO n-2:

FOR j = i+1 TO n-1:

IF employee[i].seniority > employee[j].seniority:

SWAP employee[i] and employee[j]

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| Enter the number of employees: 3  Enter employee 1  Enter ID: 001  Enter First Name: John  Enter Last Name: Doe  Enter Sex (1 for Man, 0 for Woman): 1  Enter Seniority: 20  Enter employee 2  Enter ID: 002  Enter First Name: Jane  Enter Last Name: Smith  Enter Sex (1 for Man, 0 for Woman): 0  Enter Seniority: 15  Enter employee 3  Enter ID: 003  Enter First Name: Bob  Enter Last Name: Lee  Enter Sex (1 for Man, 0 for Woman): 1  Enter Seniority: 10 | Number of Men: 2  Number of Women: 1  Sorted by Seniority:  ID: 003, Name: Bob Lee, Seniority: 10  ID: 002, Name: Jane Smith, Seniority: 15  ID: 001, Name: John Doe, Seniority: 20 |

#### Source Code

#include <iostream>

#include <string>

struct Employee {

    std::string id;

    std::string firstName;

    std::string lastName;

    int gender; //1 for men and 0 for women

    int seniority;

};

struct Department {

    Employee employee[100];

    int quantity;

};

void inputEmployeeInfo(Employee &employee) {

    std::cout << "Enter ID: ";

    std::cin >> employee.id;

    std::cout << "Enter First Name: ";

    std::cin >> employee.firstName;

    std::cout << "Enter Last Name: ";

    std::cin >> employee.lastName;

    std::cout << "Enter gender: ";

    std::cin >> employee.gender;

    std::cout<<"Enter seniority: ";

    std::cin >> employee.seniority;

}

void inputDepartment(Department &department) {

    std::cout << "Enter the number of employees: ";

    std::cin >> department.quantity;

    for (int i = 0; i < department.quantity; i++) {

        std::cout << "Enter employee " << i + 1 << "\n";

        inputEmployeeInfo(department.employee[i]);

    }

}

void countGender(Department &department, int &quantityMen, int &quantityWomen) {

    quantityMen = 0;

    quantityWomen = 0;

    for (int i = 0; i < department.quantity; i++) {

        if (department.employee[i].gender == 1) {

            quantityMen++;

        } else {

            quantityWomen++;

        }

    }

}

void sortBySeniority(Department &department) {

    for (int i = 0; i < department.quantity - 1; i++) {

        for (int j = i + 1; j < department.quantity; j++) {

            if (department.employee[i].seniority > department.employee[j].seniority) {

                Employee temp = department.employee[i];

                department.employee[i] = department.employee[j];

                department.employee[j] = temp;

            }

        }

    }

}

int main() {

    Department department;

    inputDepartment(department);

    // Count gender

    int quantityMen, quantityWomen;

    countGender(department, quantityMen, quantityWomen);

    std::cout << "Number of Men: " << quantityMen << "\n";

    std::cout << "Number of Women: " << quantityWomen << "\n";

    // Sort by seniority

    sortBySeniority(department);

    std::cout << "Employees sorted by seniority:\n";

    for (int i = 0; i < department.quantity; i++) {

        std::cout << "ID: " << department.employee[i].id

                  << ", Name: " << department.employee[i].firstName << " " << department.employee[i].lastName

                  << ", Gender: " << (department.employee[i].gender == 1 ? "Man" : "Woman")

                  << ", Seniority: " << department.employee[i].seniority << "\n";

    }

    return 0;

}

#### 2. Bài 2

#### Main Idea

BEGIN

DECLARE list[20] // Mảng chứa hoa

DECLARE count = 0 // Biến đếm số lượng hoa

FUNCTION inputFlowerInfo():

READ flower.name, flower.quantity, flower.unit, flower.price

list[count] = flower

count = count + 1

FUNCTION printFlowerList():

FOR i = 0 TO count - 1:

PRINT list[i].name, list[i].quantity, list[i].unit, list[i].price

FUNCTION searchFlowerByName(name):

FOR i = 0 TO count - 1:

IF list[i].name == name:

RETURN i

RETURN -1

FUNCTION sellFlower(name, quantity):

index = searchFlowerByName(name)

IF index != -1 AND list[index].quantity >= quantity:

total = list[index].price \* quantity

list[index].quantity -= quantity

PRINT "Total price: ", total

ELSE:

PRINT "Not enough flowers or flower not found"

DO

PRINT "1. Add Flower"

PRINT "2. Print Flower List"

PRINT "3. Sell Flower"

PRINT "0. Exit"

READ choice

IF choice == 1: CALL inputFlowerInfo()

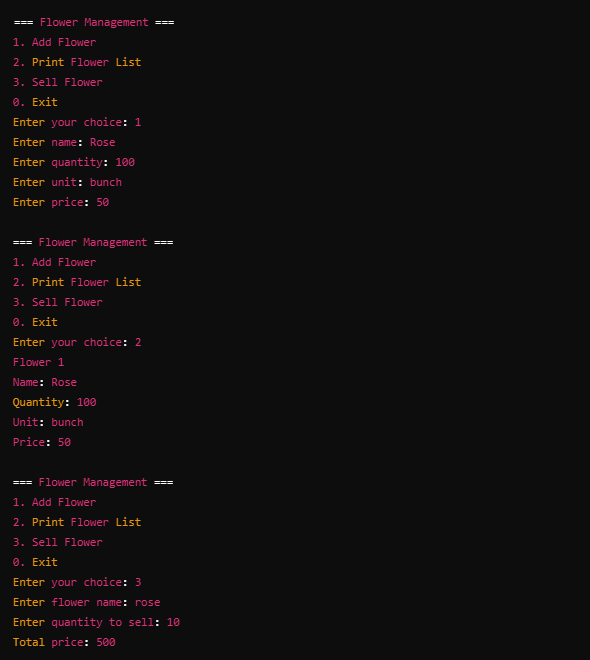
IF choice == 2: CALL printFlowerList()

IF choice == 3: CALL sellFlower(name, quantity)

WHILE choice != 0

END

#### Example – Result



#### Source Code

#include <iostream>

#include <string>

struct Flower

{

    std::string name;

    int quantity;

    std::string unit;

    int price;

    static Flower list[20];

    static int count;

};

Flower Flower::list[20];

int Flower::count = 0;

void inputFlowerInfo()

{

    if (Flower::count >= 20)

    {

        std::cout << "Flower list is full.\n";

        return;

    }

    Flower flower;

    std::cout << "Enter name: ";

    std::cin.ignore();

    std::getline(std::cin, flower.name);

    std::cout << "Enter quantity: ";

    std::cin >> flower.quantity;

    std::cout << "Enter unit: ";

    std::cin >> flower.unit;

    std::cout << "Enter price: ";

    std::cin >> flower.price;

    Flower::list[Flower::count++] = flower;

}

void printFlowerList()

{

    if (Flower::count == 0)

    {

        std::cout << "Flower list is empty.\n";

        return;

    }

    for (int i = 0; i < Flower::count; i++)

    {

        std::cout << "Flower " << i + 1 << "\n";

        std::cout << "Name: " << Flower::list[i].name << "\n";

        std::cout << "Quantity: " << Flower::list[i].quantity << "\n";

        std::cout << "Unit: " << Flower::list[i].unit << "\n";

        std::cout << "Price: " << Flower::list[i].price << "\n";

    }

}

std::string toLower(const std::string &str)

{

    std::string result;

    for (char c : str)

    {

        if (c >= 'A' && c <= 'Z')

        {

            result += c + 32; // Chuyển chữ hoa thành chữ thường

        }

        else

        {

            result += c;

        }

    }

    return result;

}

int searchFlowerByName(std::string name)

{

    std::string lowerName = toLower(name);

    for (int i = 0; i < Flower::count; i++)

    {

        if (toLower(Flower::list[i].name) == lowerName)

        {

            return i;

        }

    }

    return -1;

}

int sellingFlowerAdmin(std::string name, int quantity)

{

    int index = searchFlowerByName(name);

    if (index == -1)

    {

        std::cout << "Flower not found\n";

        return -1;

    }

    if (Flower::list[index].quantity < quantity)

    {

        std::cout << "Not enough flower\n";

        return -1;

    }

    int total = quantity \* Flower::list[index].price;

    Flower::list[index].quantity -= quantity;

    return total;

}

int main()

{

    int choice;

    do

    {

        std::cout << "\n=== Flower Management ===\n";

        std::cout << "1. Add Flower\n";

        std::cout << "2. Print Flower List\n";

        std::cout << "3. Sell Flower\n";

        std::cout << "0. Exit\n";

        std::cout << "Enter your choice: ";

        std::cin >> choice;

        switch (choice)

        {

        case 1:

            inputFlowerInfo();

            break;

        case 2:

            printFlowerList();

            break;

        case 3:

        {

            std::string name;

            int quantity;

            std::cin.ignore();

            std::cout << "Enter flower name: ";

            std::getline(std::cin, name);

            std::cout << "Enter quantity to sell: ";

            std::cin >> quantity;

            int total = sellingFlowerAdmin(name, quantity);

            if (total != -1)

            {

                std::cout << "Total price: " << total << "\n";

            }

            break;

        }

        case 0:

            std::cout << "Exiting...\n";

            break;

        default:

            std::cout << "Invalid choice. Try again.\n";

        }

    } while (choice != 0);

    return 0;

}

# IV. Lab01d

## 1. Bài 1

### Main Idea

BEGIN

FUNCTION check(str):

FOR EACH character in str:

IF character is a digit (between '0' and '9'):

RETURN false

RETURN true

// Main program

PRINT "Enter a string: "

READ input\_string

IF check(input\_string) == true:

PRINT "Khong co ki tu so."

ELSE:

PRINT "Co ki tu so."

END

### Example -Result

|  |  |
| --- | --- |
| Input | Output |
| “Hello World” | Khong co ki tu so |
| Hello123 | Co ki tu so. |
| 12345 | Co ki ty so. |
| abc!#@ | Khong co ki tu so. |

### Source Code

#include <iostream>

#include <string>

bool check(std::string str)

{

    for (char c : str)

    {

        if (c >= '0' && c <= '9')

        {

            return false;

        }

    }

    return true;

}

int main()

{

    std::string str;

    getline(std::cin, str);

    if (check(str))

    {

        std::cout << "Khong co ki tu so.";

    }

    else

    {

        std::cout << "Co ki tu so.";

    }

    return 0;

}

## 2. Bài 2

### Main Idea

BEGIN

FUNCTION check(str):

SET start = 0

SET end = length of str - 1

WHILE start < end:

IF str[start] != str[end]:

RETURN false

INCREMENT start

DECREMENT end

RETURN true

// Main program

PRINT "Enter a string: "

READ input\_string

IF check(input\_string) == true:

PRINT "La chuoi doi xung."

ELSE:

PRINT "Khong la chuoi doi xung."

END

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| “madam” | La chuoi doi xung. |
| “Hello” | Khong la chuoi doi xung. |
| “racear” | La chuoi doi xung. |
| “abcba” | La chuoi doi xung. |

### Source Code

#include <iostream>

#include <string>

bool check(std::string str)

{

    int start = 0;

    int end = str.length() - 1;

    while (start < end)

    {

        if (str[start] != str[end])

        {

            return false;

        }

        start++;

        end--;

    }

    return true;

}

int main()

{

    std::string str;

    getline(std::cin, str);

    if (check(str))

    {

        std::cout << "La chuoi doi xung.";

    }

    else

    {

        std::cout << "Khong la chuoi doi xung.";

    }

    return 0;

}

## 3. Bài 3

### Main Idea

BEGIN

FUNCTION countWords(str):

SET count = 0

SET inWord = false

FOR each character c in str:

IF c is not a space:

IF not inWord:

INCREMENT count by 1

SET inWord = true

ELSE:

SET inWord = false

RETURN count

// Main program

PRINT "Enter a string: "

READ input\_string

SET wordCount = countWords(input\_string)

PRINT "Number of words: " + wordCount

END

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| “Hello World” | Number of words: 2 |
| “ Hello World” | Number of words: 2 |
| “This is a test” | Number of words: 4 |
| “ ” | Number of words: 0 |
| “One” | Number of words: 1 |

### Source Code

#include <iostream>

#include <string>

int countWords(std::string str)

{

    int count = 0;

    bool inWord = false;

    for (char c : str)

    {

        if (c != ' ')

        {

            if (!inWord)

            {

                count++;

                inWord = true;

            }

        }

        else

        {

            inWord = false;

        }

    }

    return count;

}

int main()

{

    std::string str;

    std::getline(std::cin, str);

    int wordCount = countWords(str);

    std::cout << "Number of words: " << wordCount << std::endl;

    return 0;

}

## BT Phải làm

### 1. Bài 1

#### Main Idea

Thực hiện **sắp xếp các ký tự trong một chuỗi** theo thứ tự tăng dần dựa trên **mã ASCII**. Sử dụng thuật toán sắp xếp hoán đổi (Interchange Sort), kiểm tra từng cặp ký tự trong chuỗi và hoán đổi chúng nếu cần thiết.

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| “dcba” | “abcd” |
| “bbaacc” | “aabbcc” |
| “zyxwvutsrqponmlkjihgfedcba” | “abcdefghijklmnopqrstuvwxyz” |
| “d3 c2a1b” | “ 123abcd” |
| “z5 y2x1@#” | “#125xyz@” |

#### Source Code

#include <iostream>

#include <string>

void swap(char &a, char &b)

{

    char temp = a;

    a = b;

    b = temp;

}

void interchangeSort(std::string &str)

{

    for (int i = 0; i < str.length() - 1; i++)

    {

        for (int j = i + 1; j < str.length(); j++)

        {

            if (str[i] > str[j])

            {

                swap(str[i], str[j]);

            }

        }

    }

}

int main()

{

    std::string str;

    getline(std::cin, str);

    interchangeSort(str);

    std::cout << str;

    return 0;

}

### 2. Bài 2

#### Main Idea

Chương trình này sẽ xóa một ký tự tại một vị trí cụ thể trong chuỗi mà người dùng nhập vào. Cách thực hiện là:

1. Nhập một chuỗi từ người dùng.
2. Nhập một vị trí cụ thể mà người dùng muốn xóa ký tự tại vị trí đó.
3. Kiểm tra tính hợp lệ của vị trí (không nhỏ hơn 0 và không vượt quá chiều dài chuỗi).
4. Dịch các ký tự phía sau vị trí đó sang trái một vị trí.
5. Cắt chuỗi để bỏ ký tự cuối cùng (sau khi dịch).

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| “Hello, World!”  5 | “Hello World!” |
| “ABCDEFG”  2 | “ABDEF” |
| “OpenAI”  6 | “OpenA” |
| “Hello, World!”  20 | Invalid position! |
| “Hello”  -1 | Invalid position! |

#### Source Code

#include <iostream>

#include <string>

void removeAtPosition(std::string &str, int pos)

{

    if (pos < 0 || pos >= str.length())

    {

        std::cout << "Invalid position!";

        return;

    }

    for (int i = pos; i < str.length() - 1; i++)

    {

        str[i] = str[i + 1];

    }

    str.resize(str.length() - 1);

}

int main()

{

    std::string str;

    getline(std::cin, str);

    int pos;

    std::cin >> pos;

    removeAtPosition(str, pos);

    std::cout << str;

    return 0;

}

### 3. Bài 3

#### Main Idea

1. **Kiểm tra hợp lệ vị trí pos**: Nếu vị trí không hợp lệ (nhỏ hơn 0 hoặc lớn hơn chiều dài chuỗi), hàm sẽ thông báo lỗi và không thực hiện việc chèn.
2. **Resize chuỗi**: Tăng kích thước chuỗi thêm 1 đơn vị để tạo không gian cho ký tự mới.
3. **Dịch các ký tự**: Dịch các ký tự từ cuối chuỗi (trừ ký tự cuối cùng) về phía sau một vị trí để tạo chỗ trống.
4. **Chèn ký tự**: Gán ký tự c vào vị trí yêu cầu pos.

#### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| “Hello World!”  X  5 | “HelloX World!” |
| “OpenAI”  A  4 | “OpenAIA” |
| “Test”  T  0 | “TTest” |
| “Hello World!”  Z  50 | Invalid position! |
| “Hello”  44  -1 | Invalid position! |

#### Source Code

#include <iostream>

#include <string>

void insertCharAtPosition(std::string &str, char c, int pos)

{

    if (pos < 0 || pos > str.length())

    {

        std::cout << "Invalid position!";

        return;

    }

    str.resize(str.length() + 1);

    for (int i = str.length() - 1; i > pos; i--)

    {

        str[i] = str[i - 1];

    }

    str[pos] = c;

}

int main()

{

    std::string str;

    getline(std::cin, str);

    char c;

    int pos;

    std::cin >> c >> pos;

    insertCharAtPosition(str, c, pos);

    std::cout << str;

    return 0;

}

# V. Lab01e

## 1. Bài 1

### Source Code

#include <iostream>

#include <fstream>

#define FI "input.inp"

void inputData(int &n)

{

    std::ifstream fi(FI);

    if (!fi)

    {

        std::cerr << "Can not open the file !\n";

        exit(0);

    }

    fi >> n;

    fi.close();

}

int main()

{

    int n;

    inputData(n);

    std::cout << "n= " << n << '\n';

    return 0;

}

## 2. Bài 2

### Source Code

#include <iostream>

#include <fstream>

#define FO "output.out"

void createArray(int \*&a, int n)

{

    a = new int[n];

    for (int i = 0; i < n; i++)

    {

        std::cin >> a[i];

    }

}

void outputData(int \*a, int n)

{

    std::ofstream fo(FO);

    if (!fo)

    {

        std::cerr << "Can not create the file!" << std::endl;

        exit(0);

    }

    for (int i = 0; i < n; i++)

    {

        fo << a[i] << " ";

    }

    fo.close();

}

int main()

{

    int n;

    std::cin >> n;

    int \*a;

    createArray(a, n);

    outputData(a, n);

    delete[] a;

    return 0;

}

## BT Phải làm

### 1. Bài 1

#### Source Code

#include <iostream>

#include <fstream>

#define FI "DaySoNguyen.inp"

void inputData(int \*&a, int &n)

{

    std::ifstream fi(FI);

    if (!fi)

    {

        std::cerr << "Can not open the file !\n";

        exit(0);

    }

    fi >> n;

    a = new int[n];

    for (int i = 0; i < n; i++)

    {

        fi >> a[i];

    }

    fi.close();

}

int main()

{

    int n;

    int \*a = nullptr;

    inputData(a, n);

    for (int i = 0; i < n; i++)

    {

        std::cout << a[i] << " ";

    }

    delete[] a;

    return 0;

}

### 2. Bài 2

#### Source Code

#include <iostream>

#include <fstream>

#define FI "MangSo.inp"

void inputData(int \*\*&a, int &rows, int &cols)

{

    std::ifstream fi(FI);

    if (!fi)

    {

        std::cerr << "Can not open the file!\n";

        exit(0);

    }

    fi >> rows >> cols;

    a = new int \*[rows];

    for (int i = 0; i < rows; i++)

    {

        a[i] = new int[cols];

        for (int j = 0; j < cols; j++)

        {

            fi >> a[i][j];

        }

    }

    fi.close();

}

void outputData(int \*\*a, int rows, int cols)

{

    for (int i = 0; i < rows; i++)

    {

        for (int j = 0; j < cols; j++)

        {

            std::cout << a[i][j] << " ";

        }

        std::cout << std::endl;

    }

}

int main()

{

    int \*\*a;

    int rows, cols;

    inputData(a, rows, cols);

    outputData(a, rows, cols);

    // Giải phóng bộ nhớ

    for (int i = 0; i < rows; i++)

    {

        delete[] a[i];

    }

    delete[] a;

    return 0;

}

### 3. Bài 3

#### Source Code

#include <iostream>

#include <fstream>

#include <cmath>

#define FI "NT.inp"

#define FO "NT.out"

bool isPrime(int n)

{

    if (n < 2)

    {

        return false;

    }

    int k = sqrt(n);

    for (int i = 2; i <= k; i++)

    {

        if (n % i == 0)

        {

            return false;

        }

    }

    return true;

}

void swap(int &a, int &b)

{

    int temp = a;

    a = b;

    b = temp;

}

void interchangeSort(int \*a, int n)

{

    for (int i = 0; i < n - 1; i++)

    {

        for (int j = i + 1; j < n; j++)

        {

            if (a[i] > a[j])

            {

                swap(a[i], a[j]);

            }

        }

    }

}

void inputData(int \*&a, int &n)

{

    std::ifstream fi(FI);

    if (!fi)

    {

        std::cerr << " Can not open the file !\n";

        exit(0);

    }

    fi >> n;

    a = new int[n];

    for (int i = 0; i < n; i++)

    {

        fi >> a[i];

    }

    fi.close();

}

void outputData(int \*a, int n)

{

    std::ofstream fo(FO);

    if (!fo)

    {

        std::cerr << " Can not create the file !\n";

        exit(0);

    }

    int count = 0;

    int \*temp = new int[n];

    for (int i = 0; i < n; i++)

    {

        if (isPrime(a[i]))

        {

            temp[count++] = a[i];

        }

    }

    interchangeSort(temp, count);

    fo << count << "\n";

    for (int i = 0; i < count; i++)

    {

        fo << temp[i] << " ";

    }

    delete[] temp;

}

int main()

{

    int n;

    int \*a = nullptr;

    inputData(a, n);

    outputData(a, n);

    delete[] a;

    return 0;

}

### Example – Result

|  |  |
| --- | --- |
| Input (NT.inp) | Output (NT.out) |
| 6  10 2 3 4 5 6 | 3  2 3 5 |
| 5  11 12 13 14 15 | 2  11 13 |

### 4. Bài 4

Source Code

#include <iostream>

#include <fstream>

#include <cmath>

#define FI "CapSo.inp"

#define FO "Capso.out"

template <typename T, typename U>

struct MyPair

{

    T first;

    U second;

};

void inputData(int \*&a, int &n, int &k)

{

    std::ifstream fi(FI);

    if (!fi)

    {

        std::cerr << " Can not open the file !\n";

        exit(0);

    }

    fi >> n >> k;

    a = new int[n];

    for (int i = 0; i < n; i++)

    {

        fi >> a[i];

    }

    fi.close();

}

bool isPairExist(MyPair<int, int> \*rs, int count, int first, int second)

{

    for (int i = 0; i < count; i++)

    {

        if ((rs[i].first == first && rs[i].second == second) ||

            (rs[i].first == second && rs[i].second == first))

        {

            return true;

        }

    }

    return false;

}

void checkCondition(int \*&a, MyPair<int, int> \*&rs, int &n, int &k, int &count)

{

    count = 0;

    for (int i = 0; i < n - 1; i++)

    {

        for (int j = i + 1; j < n; j++)

        {

            if (a[i] + a[j] == k)

            {

                if (!isPairExist(rs, count, a[i], a[j]))

                {

                    rs[count].first = a[i];

                    rs[count].second = a[j];

                    count++;

                }

            }

        }

    }

}

void outputData(MyPair<int, int> \*rs, int &n)

{

    std::ofstream fo(FO);

    if (!fo)

    {

        std::cerr << "Can not create the file!" << std::endl;

        exit(0);

    }

    for (int i = 0; i < n; i++)

    {

        fo << rs[i].first << " " << rs[i].second << "\n";

    }

    fo.close();

}

int main()

{

    int n, k, count = 0;

    int \*a = nullptr;

    MyPair<int, int> \*rs;

    inputData(a, n, k);

    rs = new MyPair<int, int>[n];

    checkCondition(a, rs, n, k, count);

    outputData(rs, count);

    delete[] a;

    delete[] rs;

    return 0;

}

#### Example – Result

|  |  |
| --- | --- |
| Input (CapSo.inp) | Output (CapSo.out) |
| 5 10  1 3 2 8 7 | 3 7  2 8 |
| 6 10  1 3 3 7 2 8 | 3 7  2 8 |

# VI. Lab01f

## 1. Bài 1

### Main Idea

* Lớp Phone mô phỏng một hệ thống quản lý điện thoại. Lớp này có thể lưu trữ thông tin về điện thoại (như id, brand, price), kiểm tra sự tồn tại của điện thoại dựa trên id, thay đổi giá của điện thoại, xóa điện thoại, và hiển thị tất cả các điện thoại.
* Lớp sử dụng mảng Phone[] để lưu trữ danh sách điện thoại, có thể quản lý và thực hiện các thao tác liên quan đến việc thêm, xóa, hiển thị, và cập nhật thông tin của điện thoại.
* Phương thức static được sử dụng để lưu trữ danh sách điện thoại và số lượng điện thoại, giúp quản lý thông tin chung cho tất cả đối tượng Phone.

Các Phương thức Chính:

1. **addPhone**: Thêm một điện thoại vào danh sách nếu chưa tồn tại.
2. **showInfoPhone**: Hiển thị thông tin của điện thoại nếu tìm thấy trong danh sách.
3. **setPrice**: Cập nhật giá của một điện thoại dựa trên id.
4. **removePhone**: Xóa một điện thoại khỏi danh sách dựa trên id.
5. **display**: Hiển thị thông tin của tất cả các điện thoại trong danh sách.

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 1 “P001” “Apple” 1000  1 “P002” “Samsung” 800  2 “P001” | Phone ID: P001  Phone Brand: Apple  Phone Price: 1000 |
| 1 “P003” “OnePlus” 700  3 “P003” 750  2 “P003” | Enter new price: 750  Phone ID: P003  Phone Brand: OnePlus  Phone Price: 750 |
| 1 “P004” “Nokia” 500  4 “P004”  5 | Phone ID: P001  Phone Brand: Apple  Phone Price: 1000  ----------------------  Phone ID: P002  Phone Brand: Samsung  Phone Price: 800  ----------------------  Phone ID: P003  Phone Brand: OnePlus  Phone Price: 750  ---------------------- |
| 1 “P001” “Apple” 1000 | Phone with id P001 already exist in the system. |

## 2. Bài 2

### Main Idea

Chương trình quản lý danh sách hàng hóa trong kho hoặc cửa hàng. Mỗi mặt hàng có các thuộc tính sau:

* **ID**: Mã định danh duy nhất cho hàng hóa.
* **Tên**: Tên của hàng hóa.
* **Số lượng**: Số lượng đơn vị của hàng hóa có sẵn.
* **Giá**: Giá của mỗi đơn vị hàng hóa.

Chương trình cho phép người dùng thực hiện các chức năng sau:

1. **Thêm hàng hóa**: Thêm hàng hóa mới vào hệ thống và ngăn chặn việc trùng lặp ID.
2. **Tìm kiếm hàng hóa**: Tìm kiếm hàng hóa theo tên và hiển thị thông tin chi tiết nếu tìm thấy.
3. **Xem tổng số lượng**: Hiển thị tổng số lượng hàng hóa hiện có trong hệ thống.
4. **Quản lý tài nguyên**: Sử dụng bộ nhớ động để lưu trữ danh sách hàng hóa, đảm bảo quản lý tài nguyên hiệu quả

### Example – Result

|  |  |
| --- | --- |
| Input | Output |
| 2  “G001” “Apple” 10 50  “G002” “Orange” 20 30  “Apple” | ID: G001  Name: Apple  Quantity: 10  Price: 50  Total quantity: 30 |
| 2  “G001” “Apple” 10 50  “G002” “Orange” 20 30  “Apple” | ID G001 already exists. Skipping. |
| 1  “G00” “Mango” 20 50  “Grape” | Goods not found.  Total quantity: 10 |

### Source Code

#include <iostream>

#include <string>

class Goods

{

private:

    std::string id;

    std::string name;

    int quantity;

    int price;

    static Goods \*list;

    static int countGoods;

    static int totalQuantity;

public:

    Goods() : id(""), name(""), quantity(0), price(0) {}

    Goods(std::string id, std::string name, int quantity, int price) : id(id), name(name), quantity(quantity), price(price) { totalQuantity += quantity; }

    static void addGoods()

    {

        int n;

        std::cout << "Enter number of goods: ";

        std::cin >> n;

        for (int i = 0; i < n; i++)

        {

            std::string id, name;

            int quantity, price;

            std::cout << "Enter ID: ";

            std::cin >> id;

            for (int j = 0; j < countGoods; j++)

            {

                if (list[j].id == id)

                {

                    std::cout << "ID " << id << " already exists. Skipping.\n";

                    return;

                }

            }

            std::cin.ignore();

            std::cout << "Enter name: ";

            getline(std::cin, name);

            std::cout << "Enter quantity: ";

            std::cin >> quantity;

            std::cout << "Enter price: ";

            std::cin >> price;

            list[countGoods++] = Goods(id, name, quantity, price);

        }

    }

    static void searchGoods(std::string name)

    {

        for (int i = 0; i < countGoods; i++)

        {

            if (list[i].name == name)

            {

                std::cout << "ID: " << list[i].id << "\n";

                std::cout << "Name: " << list[i].name << "\n";

                std::cout << "Quantity: " << list[i].quantity << "\n";

                std::cout << "Price: " << list[i].price << "\n";

                return;

            }

        }

        std::cout << "Goods not found." << "\n";

    }

    static int getTotalQuantity()

    {

        return totalQuantity;

    }

};

Goods \*Goods::list = new Goods[100];

int Goods::countGoods = 0;

int Goods::totalQuantity = 0;

int main()

{

    Goods::addGoods();

    std::string name;

    std::cin.ignore();

    std::cout << "Enter name to search: ";

    getline(std::cin, name);

    Goods::searchGoods(name);

    std::cout << "Total quantity: " << Goods::getTotalQuantity() << "\n";

    return 0;

}