**Bài 1.**Cài đặt lớp tập hợp các số nguyên **SET** gồm *n* là số phần tử có trong tập hợp, *data* là mảng động chứa các phần tử của tập hợp, *size* là kích thước tối đa của tập hợp.

Lớp Set:

Các thành viên:

* Thuộc tính:

int n;

Kích thước tập hợp

int \*data;

Mảng động chứa các phần tử tập hợp

int size;

Kích thước tối đa của tập hợp

* Phương thức:

Set(int size = 20);

Hàm dựng với tham số là kích thước của tập hợp

Set(int n, int a[], int size = 20);

Hàm dựng với tham số là một mảng các số nguyên

Set(const Set &other);

Hàm sao chép

~Set();

Hàm hủy

Set operator + (int a);

Nạp chồng toán tử + thêm phần tử vào tập hợp

Set operator - (int a);

Nạp chồng toán tử - xóa một phần tử ra khỏi tập hợp

Set operator + (const Set &other);

Nạp chồng toán tử \* hợp hai tập hợp

Set operator - (const Set &other);

Nạp chồng toán tử - hiệu 2 tập hợp

Set operator \* (const Set &other);

Nạp chồng toán tử \* giao 2 tập hợp

bool operator () (int a) const;

Nạp chồng toán tử () kiểm tra một phần tử có thuộc tập hợp

int operator [] (int index);

Nạp chồng toán tử [] truy xuất một phần tử trong tập hợp

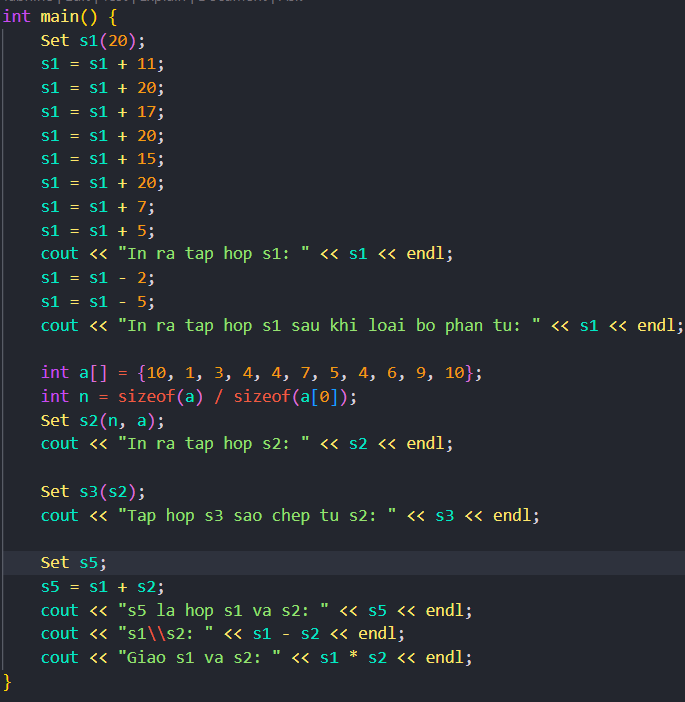
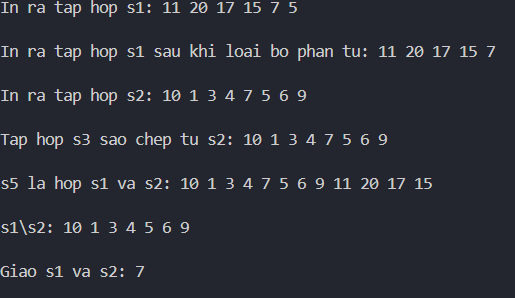
Set &operator = (const Set &other);

Nạp chồng toán tử = gán tập hợp

friend ostream &operator << (ostream &out, const Set & s);

Nạp chồng toán tử << in ra nội dung tập hợp

Hàm main, kết quả:

****

**Bài 2.**Cài đặt lớp **LinkList** là 01 danh sách liên kết (DSLK) gồm con trỏ đầu và cuối danh sách liên kết các số nguyên.

struct Node {

    int data;

    Node\* next;

};

Lớp **LinkList** :

Các thành viên:

* Thuộc tính:

Node\* head;

Kích thước tập hợp

Node\* tail;

Mảng động chứa các phần tử tập hợp

* Phương thức:

LinkList();

Hàm dựng với tham số là kích thước của tập hợp

LinkList(LinkList &other);

Hàm sao chép

~LinkList();

Hàm hủy

void insert\_front(int x);

Thêm một phần tử vào đầu danh sách

void insert\_back(int x);

Thêm một phần tử vào cuối danh sách

void insert\_middle(Node\* position, int x);

Thêm một phần tử sau một phần tử position

void delete\_front();

Xóa một phần tử vào đầu danh sách

void delete\_back();

Xóa một phần tử vào cuối danh sách

void delete\_middle(Node\* position);

Xóa một phần tử sau một phần tử position

friend ostream &operator << (ostream &out, const LinkList & L);

Nạp chồng toán tử << in ra nội dung tập hợp

void sort(bool CompFunction(int, int));

Hàm sắp xếp các phần tử trong DSLK tăng/giảm dần

Hàm main, kết quả:

