

DATA 1

Abed Alkareem Hourani



 **KASIT ACADIMIC TEAM**

 **KASIT TEAM**

 **KASIT_TEAM**

 **KASIT TEAM**

 **KASIT_TEAM**

 **WWW.KASIT.NET**



Data1

Struct:

```
#include<iostream>
#include<string>
using namespace std;
struct student
{
    int id;
    string first , last;
    int greads[3];
    double avg;
};
void main()
{
    student s1;
    cin>>s1.first>>s1.last;
    cin>>s1.id;
    int sum=0;
    for(int i=0 ; i<3 ; i++)
    {
        cin>>s1.greads[i];
        sum=sum+s1.greads[i];    // sum+=s1.greads[i];
    }
    s1.avg=sum/3.0;

    cout<<s1.first<<" "<<s1.last<<endl;
    cout<<s1.id<<endl;
    for(int i=0 ; i<3 ; i++)
        cout<<s1.greads[i]<<" ";
    cout<<endl;
    cout<<s1.avg<<endl;
}
```

```

struct student
{
    int id;
    string first , last;
    int grades[3];
    double avg;
};

```

عرفنا struct باسم student يحتوي على

int id : نعني فيها رقم الجامعي للطلاب من نوع

String first , last : نعني فيها الاسم الاول و الثاني للطلاب من نوع

int grades[n] : نعني فيها علامات الطلاب من نوع array & int , n : عدد العلامات (حجم array)

Double avg : نعني فيها معدل الاعلامات من نوع double

طريقة تعريف ال struct في ال main

```

void main()
{
    student s1;
    cin>>s1.first>>s1.last; // جعل المستخدم يدخل الاسم الاول و الثاني
    cin>>s1.id; // جعل المستخدم يدخل رقم الجامعي
    int sum=0; // اعطاء قيمة اولية
    for(int i=0 ; i<3 ; i++)
    {
        cin>>s1.greads[i];
        sum=sum+s1.greads[i]; // طريقة اخرى لتعريف
    }
    s1.avg=sum/3.0;

    cout<<s1.first<<" "<<s1.last<<endl;
    cout<<s1.id<<endl;
    for(int i=0 ; i<3 ; i++)
        cout<<s1.greads[i]<<" ";
    cout<<endl;
    cout<<s1.avg<<endl;
}

```

s1 student : قمنا بتعرف object من نوع struct student باسم

كل اشي معرف داخل ال struct اصبح معرف داخل s1 object

كيف نستدعي الاوامر الموجودة في struct student من خلال ال s1 object ؟

(اسم الامر الذي نريده) .s1

```
for(int i=0 ; i<3 ; i++)
{
    cin>>s1.greads[i];
    sum=sum+s1.grades[i];
}
```

عشان عرفنا لازم نستخدم for loop حتى ندخل علامات طالب

عملنا $sum=sum+s1.grades[i]$ لانه $avg = \text{مجموع العلامات} / \text{عدداهم (n)}$ قمنا بجمع العلامات خلال ادخال المستخدم لهم

اكتب الكود الان حتى تستفيد

لو بدى اعبي اكثر من شخص ؟ يستخدم ال array لمعادات ال struct

```
struct student {
    int stno;
    string first, last;
    int grades[3];
    double avg;
};

void main() {
    student st[3];

    for (int i = 0; i < 3; i++)
    {
        cout << "please enter student" << i << "number";
        cin >> st[i].stno;
        cout << "please enter student" << i << "name";
        cin >> st[i].first >> st[i].last;

        for (int g = 0; g < 3; g++)
        {
            int sum = 0;
            cout << "please enter student" << i << "grades";
            cin >> st[i].grades[g];
            sum += st[i].gardses[g];
        }

        st[i].avg = sum / 3;
        cout << "student" << i + 1 << "info";
        cout << st[i].stno << endl;
        cout << st[i].first << st[i].last << endl;
        cout << st[i].avg << endl;
    }
}
```


Struct with struct :

```
#include<iostream>
#include<string>
using namespace std;
struct nametype
{
    string first,last;
};
struct student
{
    int id;
    nametype name;
    int grades[3];
    double avg;
};
void main()
{
    student s1[15];

    for(int j=0;j<2;j++)
    {
        int sum=0;
        cout<<"please enter id,first,last:"<<endl;
        cin>>s1[j].id>>s1[j].name.first>>s1[j].name.last;

        cout<<"please enter 3 grades:"<<endl;
        for(int i=0;i<3;i++)
        {
            cin>>s1[j].grades[i];
            sum+=s1[j].grades[i];
        }

        s1[j].avg=sum/3.0;

        cout<<"id="<<s1[j].id<<endl;
        cout<<"first="<<s1[j].name.first<<endl;
        cout<<"last="<<s1[j].name.last<<endl;

        cout<<"3 grades : "<<endl;
        for(int i=0;i<3;i++)
            cout<<s1[j].grades[i]<<endl;

        cout<<"avg="<<s1[j].avg<<endl;
    }
}
```

اكتب الكود الان حتى تستفيد

Member & non_member function :

1-non_member function :

```
#include<iostream>
#include<string>
using namespace std;
struct student
{
    int id;
    string first , last;
    int grades[3];
    double avg;
};

void read(student & s1) // الفنكشن الاول لادخال المعلومات
{
    cin>>s1.first>>s1.last;
    cin>>s1.id;
    int sum=0;
    for(int i=0 ; i<3 ; i++)
    {
        cin>>s1.grades[i];
        sum=sum+s1.grades[i]; // sum+=s1.greads[i];
    }
    s1.avg=sum/3.0;
}

void print(student s1) // الفنكشن الثاني لطباعة المتغيرات
{
    cout<<s1.first<<" "<<s1.last<<endl;
    cout<<s1.id<<endl;
    for(int i=0 ; i<3 ; i++)
        cout<<s1.grades[i]<<" ";
    cout<<endl;
    cout<<s1.avg<<endl;
}

void main()
{
    student s[10];
    for(int i=0 ; i<3 ; i++)
    {
        read(s[i]);
        print(s[i]);
    }
}
```

void read(student & s1)

حتى نربط ال struct student مع الفنكشن الاول وضعنا داخل الفنكشن student & s1 و من خلال ذلك نستطيع استدعي الاوامر الموجوده داخل struct

لماذا اشارة & ؟

بسبب حدوث تغير في المتغيرات (يجب وضعها)

مثلا رح يستدعي id و بعدين يعطيها قيمه

```
void print(student s1)
```

حتى نربط ال struct student مع الفنكشن الثاني وضعنا داخل الفنكشن student s1 و من خلال ذلك نستطيع استدعى الاوامر الموجوده داخل struct

لماذا لم نستخدم اشاره & ؟

بسبب عدم حدوث تغير على المتغيرات (لا يوجد داعي لها هنا)

اكتب الكود الان حتى تستفيد

2-member function:

```
#include<iostream>
#include<string>
using namespace std;
struct student
{
    int id;
    string first , last;
    int grades[3];
    double avg;
    void read();
    void print();
};
void student:: read()
{
    cin>>first>>last;
    cin>>id;
    int sum=0;
    for(int i=0 ; i<3 ; i++)
    {
        cin>>grades[i];
        sum=sum+grades[i];    // sum+=grades[i];
    }
    avg=sum/3.0;
}
void student:: print()
{
    cout<<first<<" "<<last<<endl;
    cout<<id<<endl;
    for(int i=0 ; i<3 ; i++)
        cout<<grades[i]<<" ";
    cout<<endl;
    cout<<avg<<endl;
}
void main()
{
    student s[10];
    for(int i=0 ; i<3 ; i++)
    {
        s[i].read();
        s[i].print();
    }
}
```



```
struct student
{
    int id;
    string first , last;
    int greads[3];
    double avg;
    void read();
    void print();
};
```

void read : فنكشن لادخال المعلومات

void print : فنكشن لطباعة المعلومات

```
void student:: read()
void student:: print()
```

طريقة تعرف member function

نوع الفنكشن (void) اسم struct (student) (:) اسم الفنكشن (read/print)

```
{
    //كود الفنكشن (عمل الفنكشن)
}
```

الفرق بين member & non_member function

Member	Non_member
يعرف داخل struct	لا يعرف داخل struct
طريقة ربط مثال void student ::read	طريقة ربط مثال read(student & s1)

اكتب الكود الان حتى تستفيد

Set&get:

Header:

```
#include<iostream>
#include<string>
using namespace std;
class student
{
//private:
    int id;
    string first , last;
public:
    void read();
    void print();
    void set(int i , string f , string l);
    void get(int &i , string &f , string &l);
    bool compaer(const student & s2);
};
```


Imp:

```
#include<iostream>
#include<string>
#include"student.h"
using namespace std;
void student:: read()
{
    cin>>first>>last;
    cin>>id;
}
void student:: print()
{
    cout<<first<<" "<<last<<endl;
    cout<<id<<endl;
}
void student::set(int i , string f , string l)
{
    id=i;
    first=f;
    last=l;
}
void student::get(int &i , string &f , string &l)
{
    i=id;
    f=first;
    l=last;
}
bool student::compaer(const student & s2 )
{
    return id==s2.id;
}

Main:
#include<iostream>
#include<string>
#include"student.h"
using namespace std;
void main()
{
    int a;
    string c,b;
    student s , s2;
    s.set(166250 , "abed al kareem" , "hourani");
    s2.read();

    if(s.compaer(s2))
        cout<<"the same student"<<endl;
    else
        cout<<"not the same"<<endl;
}
```

```

class student
{
//private:
    int id;
    string first , last;
public:
    void read();
    void print();
    void set(int i , string f , string l);
    void get(int &i , string &f , string &l);
    bool compaer(const student & s2);
};

```

الكلاس دائما يكون private بشأن هيك لازم ابين الاشياء الي بندي باها من خلال كلمة public

Header:

```
void set(int i , string f , string l);
```

يقوم الفنكشن بتخزين رقم الجامعي (i) و اسم الاول (f) و الاسم الثاني (l) حسب ترتيب اذا لم يتم الاندخال حسب ترتيب بيعطيني error اذا كان من نوع ثاني (عندي int ويدخل string) او خطأ في المعلومات اذا من نفس نوع لكن الدخال مش على ترتيب (ادخل الاسم الثاني ثم الاسم الاول رح يطبع (الاسم ثاني الاول) بدل من (الاول الثاني))

لم اضع اشارة & لعدم حدوث تغير لان المستخدم يدخل القيم مباشرة

```
void get(int &i , string &f , string &l);
```

يقوم الفنكشن باسترجاع القيم الموجوده في الذاكرة

وضعتنا & بسبب حدوث تغير في القيم و لانه رح يسترجع القيمة

مثال سوف يخزن قيمة id في i ثم سوف يرجع هاي القيمة

```
bool compaer(const student & s2);
```

يقوم الفنكشن بالمقارنة بين 2 object

وضعتنا const حتى لا يحدث تغير على قيمتها بعد الانتهاء من الفنكشن

اكتب الكود الان حتى تستفيد

Constructor:

Header:

```

#include<iostream>
#include<string>
using namespace std;
class student
{
//private:
    int id;
    string first , last;
public:
    void read();
    void print();
    void set(int i , string f , string l);
    void get(int &i , string &f , string &l);
    bool compaer(const student & o);
    student(int i=0 , string f=" " , string l=" ");
};

```

Imp:

↓
نفس الي قبل +

```

student::student(int i , string f , string l)
{
    /*      id=i;
            first=f;
            last=l;          */
            set(i,f,l);
}
Main:
#include<iostream>
#include<string>
#include"student.h"
using namespace std;
void main()
{
    student s1 , s2(166255 , "ahmad" , "hourani") , s3(166444 ) , s4(166555 , "abed")
    /*, s5("abed" ,"hourani")*/;
    s1.set(166525 , "abed al kareem " , "hourani");
    s1.print();
    s2.print();
    s3.print();
    s4.print();
}

```

Header:

```
student(int i=0 , string f=" " , string l=" ");
```

عرفنا ال constructor و اعطينا قيم (متغيرات المعرفة داخل الكلاس) و اعطيناها قيم اولية هيك بصير عندي

constructor with default parameter

imp:

```

student::student(int i , string f , string l)
{
    /*      id=i;
            first=f;
            last=l;          */
            set(i,f,l);
}

```

↑ ال constructor بعمل نفس عمل set يعني نفس الكود او اذا كان مبني عندي فنكشن ال set بنادي فنكشن وبس مثل

شرح ال main على الكود لكتب

اكتب الكود الان حتى تستفيد

Inheritance:

Header:

Student

```
class student :public person
```

```
{ //private:
```

```
    int id;
```

```
    string first , last;
```

```
    int greads[3];
```

```
    double avg;
```

```
public:
```

```
    void read();
```

```
    void print();
```

```
    void set(int i , string f , string l , int sn , string nat);
```

```
    void get(int &i , string &f , string &l);
```

```
    bool compaer(const student &o);
```

```
    student(int i=0 , string f=" " , string l=" " , int sn=0 , string nat=" ");
```

```
};
```

person

```
class person
```

```
{
```

```
//private:
```

```
    int ssno;
```

```
    string nathionalty;
```

```
public:
```

```
    void read();
```

```
    void print();
```

```
    void set(int sn , string nat);
```

```
    person(int sn=0 , string nat=" ");
```

```
};
```

imp:

Student

```
void student:: read()
```

```
{
```

```
    cin>>first>>last;
```

```
    cin>>id;
```

```
    int sum=0;
```

```
    for(int i=0 ; i<3 ; i++)
```

```
    {
```

```
        cin>>greads[i];
```

```
        sum=sum+greads[i];    // sum+=greads[i];
```

```
    }
```

```
    avg=sum/3.0;
```

```
    person::read();
```

```
}
```

```
void student:: print()
```

```
{
```

```
    cout<<first<<" "<<last<<endl;
```

```
    cout<<id<<endl;
```

```
    for(int i=0 ; i<3 ; i++)
```

```
        cout<<greads[i]<<" ";
```

```
    cout<<endl;
```

```
    cout<<avg<<endl;
```

```
    person::print();
```

```
}
```

```
void student::set(int i , string f , string l , int sn , string nat)
{
    id=i;
    first=f;
    last=l;
    for(int i=0 ; i<3 ; i++)
        greads[i]=0;
    avg=0;
    person::set(sn,nat);
}
void student::get(int &i , string &f , string &l)
{
    i=id;
    f=first;
    l=last;
}
bool student::compaer(const student &o )
{ return id==o.id; }
student::student(int i , string f , string l , int sn , string nat)
{ set(i,f,l ,sn,nat) ; }
```

```
person
void person::read()
{
    cout<<"enter your ssno"<<endl;
    cin>>ssno;
    cout<<"enter your nathonalty"<<endl;
    cin>>nathonalty;
}
void person::print()
{
    cout<<ssno<<endl;
    cout<<nathonalty<<endl;
}
void person::set(int sn , string nat)
{
    ssno=sn;
    nathonalty=nat;
}
person::person(int sn , string nat)
{
    set(sn,nat);
}
```

```
main:
void main()
{
    student s1 ,s2, s3(166250 , "abed al kareem" , "hourani" );
    s1.read();
    s1.print();
    cout<<"/////////////////////"<<endl;
    s2.set(166250 , "abed al kareem" , "hourani" , 995858584 , "jor" );
    s2.print();
    s3.print();
}
```

طريقة الربط في Inheritance تعريف الكلاس وتحديد نوع العلاقة (private, public, protected) مثل الشكل الي تحت

```
class student :public person
```

قمنا بربط كلاس person مع كلاس student و حددنا العلاقة private (يجب وضع :)

```
person::read();
```

```
person::print();
```

```
person::set(sn,nat);
```

بعد ما ربطنا الكلاس لازم اعرف الاشياء الموجودة في كلاس person في كلاس student (يجب تعديل في set, constructor, get في header و تعديل في (set, get, constructor, read, print) في imp

ملاحظة: تعديل مثل شرط يس على جدول جدول الموجودين في المثال ممكن يكون في اشياء كمان لازم اعدلها و في اشياء ممكن ما اعدلها و موجودة عندي مثل ال get في اغلب الاوقات ما بتقيني ما بعدل عليها عند ربط.

مثال ثاني :

Header:

```
#ifndef person_h
#define person_h
#include<string>
using namespace std;
class person
{
private:
    int ssno;
    string nathionalty;
protected:
    char marrid;
public:
    void read();
    void print();
    void set(int sn , string nat,char m );
    person(int sn=0 , string nat=" " ,char m=' ');
};
#endif

class student :public person
{
//private:
    int id;
    string first , last;
    int greads[3];
    double avg;
public:
    void read();
    void print();
    void set(int i , string f , string l , int sn , string nat , char m);
    void get(int &i , string &f , string &l);
    bool compaer(const student &o);
    student(int i=0 , string f=" " , string l=" " , int sn=0 , string nat=" " , char m=' ');
};
```


Imp:

```

void person::read()
{
    cout<<"enter your ssno"<<endl;
    cin>>ssno;
    cout<<"enter your nathonalty"<<endl;
    cin>>nathonalty;
    cout<<"enter if you marrid or not"<<endl;
    cin>>marrid;
}

void person::print()
{
    cout<<ssno<<endl;
    cout<<nathonalty<<endl;
    cout<<marrid<<endl;
}

void person::set(int sn , string nat , char m)
{
    ssno=sn;
    nathonalty=nat;
    marrid=m;
}

person::person(int sn , string nat , char m )
{
    set(sn,nat,m);
}

void student:: read()
{
    cin>>first>>last;
    cin>>id;
    int sum=0;
    person::read();
    /*cout<<"enter if you marrid or not "<<endl;
    cin>>marrid;*/
}

void student:: print()
{
    cout<<first<<" "<<last<<endl;
    cout<<id<<endl;
    person::print();
    //cout<<marrid<<endl;
}

void student::set(int i , string f , string l , int sn , string nat , char m )
{
    id=i;
    first=f;
    last=l;
    person::set(sn,nat,m);
    //marrid=m;
}

void student::get(int &i , string &f , string &l)
{
    i=id;
    f=first;
    l=last;
}

```

```
bool student::compaer(const student & o )
{
    return id==o.id;
}
student::student(int i , string f , string l , int sn , string nat , char m )
{
    set(i,f,l , sn , nat , m);
}
```

Main:

```
void main()
{
    student s1 , s2 , s3(166250 , "abed al kareem" , "hourani" , 995858584 );
    /*s1.read();
    s1.print();*/
    cout<<"//////////////////"<<endl;
    s2.set(166250 , "abed al kareem" , "hourani" , 995858584 , "jor" , 'n');
    s2.print();
    s3.print();
}
```

نقطة اخرى:

Valid – inValid

Header:

```
class student : public person
{
private:
    int a;
public:
    int b;
    void read();
protected:
    int c;
};
```

```
class person
{
private:
    int x;
public:
    int y;

protected:
    int z;
};
```

Imp:

```
void student::read()
{
    a;
    b;
    c;
    //-----//
    x;
    y;
    z;
}
```

Main:

```
void main()
{
    student s;
    // ---
    s.a; // a is private
    s.b; // b is public
    s.c; // c is protected
    // ---
    //-----//
    // ---
    s.x; // x is private
    s.y; // y is public
    s.z; // z is protected
    // ---
}
```

student prototype

persone prototype

جرب الكود و غير العلاقة و شوف شو المسموح و شو لا

باختصار:

Imp: شو ما كانت العلاقة (private,public,protected) مسموح نادي public , protected
valid اما ال private لا invalid من الفنكشن الجديد (person)

اما الفنكشن نفسه (student) ممكن نادي شو ما كانت العلاقة لانه من نفس ال valid class

Main: اذا العلاقة private او protected ما بقدر استدعي اشئ من الفنكشن الجديد شو ما كان نوعه
invalid x,y,z يعني pr,pu,pro

اما اذا كانت العلاقة public بقدر استدعي الاشئ نوعه public من الفنكشن الجديد يعني y - valid x , z - invalid

الفنكشن نفسه ما بقدر استدعي منه في ال main غير الاشئ الي نوعه public فقط

inh. valid b - invalid a,c شو ما كانت علاقة ال

نوع العلاقة	public	private	protected
imp	Valid public,protected	Valid public,protected	Valid public,protected
Main	Valid public	Invalid all	Invalid all

Composition:

Header:

```
class person
{
//private:
    int ssno;
    string nathionalty;
public:
    void read();
    void print();
    void set(int sn , string nat);
    person(int sn=0 , string nat=" ");
};
```

```
class student :public person
{
//private:
    int id;
    string first , last;
    int greads[3];
    double avg;
    date bd;
public:
    void read();
    void print();
    void set(int i , string f , string l , int sn , string nat , int d , int m , int
y);
    void get(int &i , string &f , string &l);
    bool compaer(const student & o);
    student(int i=0 , string f=" " , string l=" " , int sn=0 , string nat=" " , int
d=1 , int m=1 , int y=1900);
};
Imp:
```

```
void person::read()
{
    cout<<"enter your ssno"<<endl;
    cin>>ssno;
    cout<<"enter your nathionalty"<<endl;
    cin>>nathionalty;
}
void person::print()
{
    cout<<ssno<<endl;
    cout<<nathionalty<<endl;
}
void person::set(int sn=0 , string nat=" ")
{
    ssno=sn;
    nathionalty=nat;
}
person::person(int sn , string nat)
{
    set(sn,nat);
}
```

```
class date
{
    int day, mounth , year;

public:
    void read();
    void print();
    void set(int d,int m,int y);
    date(int d=1 , int m=1 , int
y=1900);
};
```

```

void date::read()
{
    cout<<"enter your birthdate"<<endl;
    cin>>day>>mounth>>year;
}
void date::print()
{
    cout<<day<<"/"<<mounth<<"/"<<year<<endl;
}
void date::set(int d , int m , int y)
{
    day=d;
    mounth=m;
    year=y;
}
date::date(int d , int m , int y)
{
    set(d,m,y);
}

void student:: read()
{
    cin>>first>>last;
    cin>>id;
    int sum=0;
    person::read();
    bd.read();
}
void student:: print()
{
    cout<<first<<" "<<last<<endl;
    cout<<id<<endl;
    person::print();
    bd.print();
}
void student::set(int i , string f , string l , int sn , string nat, int d, int m , int y)
{
    id=i;
    first=f;
    last=l;
    person::set(sn,nat);
    bd.set(d,m,y);
}
void student::get(int &i , string &f , string &l)
{
    i=id;
    f=first;
    l=last;
}
bool student::compaer(const student & o )
{
    return id==o.id;
}
student::student(int i , string f , string l , int sn , string nat , int d , int m , int y)
{
    set(i,f,l , sn , nat , d , m , y); }
    
```

Main:

```
void main()
{
    student s1 , s2;
    s1.read();
    s1.print();
    s2.set(166250 , "abed al kareem " , "hourani" , 99585475 , "jon" , 24 , 8 , 1998);
    s2.print();
}
```

الاختلاف عن الكود الي قبل (Inheritance) موجود عنا كلاس date و بننا نربطه بس مش من خلال Inheritance من خلال composition يعرف اسم الكلاس الجديد (date) داخل الكلاس القديم (student) كا متغير عادي و يعطي اسم (object) ونفس الاشي بعدل نفس ال Inheritance يعني لازم اعرف الاشياء الموجودة في كلاس date في كلاس student (يجب تعديل في set, constructor, get في header و تعديل في (set , get, constructor, read, print) في imp ملاحظة: تعديل مش شرط بس على هذول هذول الموجودين في المثال ممكن يكون في اشياء كمان لازم اعدلها و في اشياء ممكن ما اعدلها و موجودة عندي مثل ال get في اغلب الارقات ما بتقيني ما بعدل عليها عند ربط.

شرح في الكود

Test bank:

Header:

Date:

```
//avoid multiple inclusion
#ifndef date_h
#define date_h
class date
{
public:
    void read();
    void print() const;
    date(int d = 0, int m = 1, int y = 1000);
    int get_year() const;
    void set_date(int d, int m, int y);
    void get_date(int &d, int &m, int &y);
private:
    int day, month, year;
};
#endif
```

Person:

```
#ifndef _person
#define _person
#include <string>
//include suitable header file
#include "date.h"
using namespace std;
class person
{
public:
    void read();
    void print() const;
    person(int sn = 0, string nat = "", int bd = 1, int bm = 1, int by = 1000);
    int cal_age() const; // calculate the age of the person
    void set(int sn, string nat, int bd, int bm, int by);
    void get(int &sn, string &nat, int &bd, int &bm, int &by);
protected:
    char learner; // meaning does the person is educated or not y or n ?
    string blood_group;
    int age;
private:
    int ssno; // abbreviation to social security number
    string nationality;
    date birthdate;
};
#endif
```

Student:

```
#include <string>
#include "person.h"
using namespace std;
```

```
//student is a person
class student :public person
{
public:
    void read();
    void print() const;
    void set(int s, string f, string l);
    void get(int &s, string &f, string &l);
    bool compare(const student &other) const;
    student(int sn = 0, string nat = "", string bg = "", int s = 0, string f = "",
string l = "", char le = ' ', int bd = 1, int bm = 1, int by = 1000, int ed = 1, int em =
1, int ey = 2016);
    void getall(int &sn, string &nat, string &bg, int &s, string &f, string &l, char
&le, int &bd, int &bm, int &by, int &ed, int &em, int &ey);
    void setall(int sn, string nat, string bg, int s, string f, string l, char le, int
bd, int bm, int by, int ed, int em, int ey);
    void get_exam_date(int &ed, int &em, int &ey);
private:
    int stno;
    string first, last;
    int grades[3];
    double avg;
    //student has a exam_date
    date exam_date;
};
```

Imp:

Date:

```
#include <iostream>
#include "date.h"
using namespace std;
void date::read()
{
    cout << "please enter date(day, month, year): " << endl;
    cin >> day >> month >> year;
}
void date::print() const
{
    cout << day << "/" << month << "/" << year << endl;
}
date::date(int d, int m, int y)
{
    day = d;
    month = m;
    year = y;
}
int date::get_year() const
{
    return year;
}
void date::set_date(int d, int m, int y)
{
    day = d;
    month = m;
    year = y;
}
```

```
void date::get_date(int &d, int &m, int &y)
{
    d = day;
    m = month;
    y = year;
}
```

Person:

```
#include "person.h"
#include <iostream>
using namespace std;
// define the set function for the given parameters
void person::set(int sn, string nat, int bd, int bm, int by)
{
    ssno=sn;
    nationality=nat;
    birthdate.set_date(bd,bm,by);
}

// define the get function for the given parameters
void person::get(int &sn, string &nat, int &bd, int &bm, int &by)
{
    sn=ssno;
    nat=nationality;
    birthdate.get_date(bd,bm,by);
}

// define the cal_age function that subtract the birthdate year from current year
int person::cal_age()const
{
    int y=birthdate.get_year();
    return 2017-y;
}

void person::read()
{
    cout << "enter ssno, nat: " << endl;
    cin >> ssno >> nationality;
    cout << "enter birth date" << endl;
    birthdate.read();
}

void person::print()const
{
    cout << "ssno= " << ssno << endl;
    cout << "nationality:" << nationality << endl;
    cout << "birth date: " << endl;
    birthdate.print();
    cout << "age= " << cal_age() << endl;
}

person::person(int sn, string nat, int bd, int bm, int by)
{
    ssno = sn;
    nationality = nat;
}
```


Student:

```

void student::read()
{
    int sum = 0;
    person::read();
    cout << "enter blood group: " << endl;
    cin >> blood_group;
    cout << "enter stno, first, last, 3 grades: " << endl;
    cin >> stno >> first >> last;
    for (int i = 0; i < 3; i++)
    {
        cin >> grades[i];
        sum += grades[i];
    }
    avg = sum / 3.0;
    cout << "enter learner or not y/n?" << endl;
    cin >> learner;
    cout << "enter exam date: " << endl;
    exam_date.read();
}

void student::set(int s, string f, string l)
{
    stno = s;
    first = f;
    last = l;
}

void student::get(int &s, string &f, string &l)
{
    s = stno;
    f = first;
    l = last;
}

// define the set_all function for the given parameters
void student::setall(int sn, string nat, string bg, int s, string f, string l, char le,
int bd, int bm, int by, int ed, int em, int ey)
{
    person::set(sn,nat,bd,bm,by);
    stno=s;
    first=f;
    last=l;
    learner=le;
    blood_group=bg;
    exam_date.set_date(ed,em,ey);
}

// define the get_all function for the given parameters
void student::getall(int &sn, string &nat, string &bg, int &s, string &f, string &l, char
&le, int &bd, int &bm, int &by, int &ed, int &em, int &ey)
{
    person::get(sn,nat,bd,bm,by);
    s=stno;
    f=first;
    l=last;
    le=learner;
    bg=blood_group;
    exam_date.get_date(ed,em,ey);
}

} //getall

```

```

// define the get_exam_date function for the given parameters
void student::get_exam_date(int &ed, int &em, int &ey)
{
    exame_date.get_date(ed,em,ey);
}

//get_exam_date

bool student::compare(const student &other) const
{
    return stno == other.stno;
}

student::student(int sn, string nat, string bg, int s, string f, string l, char le, int
bd, int bm, int by, int ed, int em, int ey)
{
    set(s, f, l);
    for (int i = 0; i < 3; i++)
        grades[i] = 0;
    avg = 0;
    blood_group = bg;
    learner = le;
}

Main:
void main()
{
    //define 4 young students
    student young_s[4];

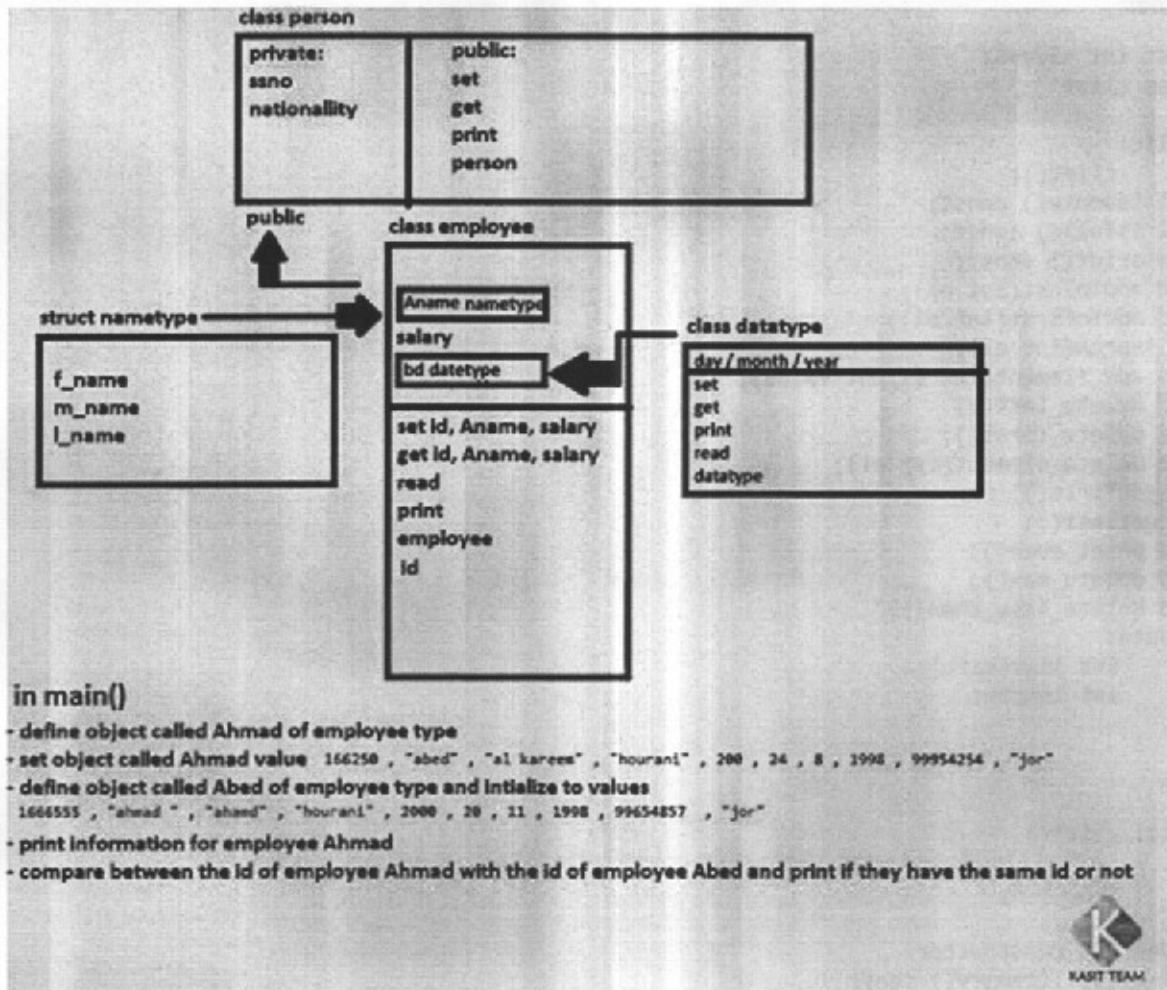
    young_s[0].setall(111, "aaa", "a", 12345, "ahmad", "khaled", 'y', 4, 3, 1997, 20,
7, 2016);
    young_s[1].setall(222, "bbb", "b", 1545, "ahmed", "khal", 'y', 6, 5, 1997, 1, 8,
2015);
    young_s[2].setall(333, "bcb", "o", 11245, "hmed", "khl", 'y', 6, 5, 1997, 8, 8,
2016);
    young_s[3].setall(343, "bcf", "ab", 1145, "hamed", "khlil", 'y', 5, 5, 1996, 20,
7, 2016);

    //print the age for each student
    for(int i=0 ; i<4 ; i++)
    {
        cout<<young_s[i].cal_age();
    }

    // count how many students that have exam at the same date with your exam date
    (20/7/2016)
    int count=0;
    for(int i=0 ; i<4 ; i++)
    {
        int d,m,y;
        young_s[i].get_exam_date(d,m,y);
        if (d==20)
            if(m==7)
                if(y==2016)
                    count++;
    }
}

```


اختبر نفسك :



الحل موجود على قناة الكاسيت



List:

Header:

```

const int size=5;
class clist
{
public:
    clist();
    bool isempty() const;
    bool isfull() const;
    void print() const;
    void addtolast(int el);
    void addtofirst(int el);
    int search(int el);
    void add_element(int el,int value);
    void delete_last();
    void delete_first();
    void delete_element(int el);
    int getfirst();
    int getlast();
    void print_even();
    void delete_max();
    void delete_less_than();
private:
    int list[size];
    int length;
};

Imp:

clist::clist()
{
    length=0;
}

//default constructor
bool clist::isempty() const
{
    return length==0;
}

//isempty
bool clist::isfull() const
{
    return length==size;
}

//isfull
void clist::print() const
{
    if(!isempty())
    {
        for(int i=0;i<length;i++)
            cout<<list[i]<<" ";
        cout<<endl;
    }
    else
        cout<<"sorry list is empty"<<endl;
}

//print

```

List:

Header:

```
const int size=5;
class clist
{
public:
    clist();
    bool isempty() const;
    bool isfull() const;
    void print() const;
    void addtolast(int el);
    void addtofirst(int el);
    int search(int el);
    void add_element(int el,int value);
    void delete_last();
    void delete_first();
    void delete_element(int el);
    int getfirst();
    int getlast();
    void print_even();
    void delete_max();
    void delete_less_than();
private:
    int list[size];
    int length;
};

Imp:

clist::clist()
{
    length=0;
}

//default constructor
bool clist::isempty() const
{
    return length==0;
}

//isempty
bool clist::isfull() const
{
    return length==size;
}

//isfull
void clist::print() const
{
    if(!isempty())
    {
        for(int i=0;i<length;i++)
            cout<<list[i]<<" ";
        cout<<endl;
    }
    else
        cout<<"sorry list is empty"<<endl;
}

//print
```

array →

التأكد أو التخصيص بناءً على أن فارغة

التأكد أو التخصيص بناءً على أن مليئة

→ For List

ههه



اضافة عنصر في آخر List

```
void clist::addtolast(int el)
```

```
{
    if(!isfull()) → التأكد اذا كانت List مليئة لا يضيف
        list[length++] = el;
    else → اما اذا كانت غير ممتلئة فانه يضيف على الـ List
        cout << "sorry list is full" << endl;
} //addtolast
```

(length++) نستعمل فيه Length ثم نزيد

```
void clist::addtofirst(int el)
```

```
{
    if(!isfull()) → التأكد اذا كنا ممتلئين ام لا
    {
        for(int i=length; i>0; i--)
            list[i] = list[i-1];

        list[0] = el;
        length++; → لزيادة طول List
    }
    else → لاضافة عنصر جديد
        cout << "sorry list is full" << endl;
} //addtofirst
```

اضافة عنصر في بداية List

شرح: انقل كل عنصر للمكان الذي يليه

```
int clist::search(int el)
```

```
{
    int pos = -1;
    if(!isempty()) → التأكد اذا List فارغ
    {
        for(int i=0; i<length; i++)
            if(list[i] == el)
            {
                pos = i;
                break;
            } //if
    }
    else
        cout << "sorry list is empty" << endl;
}
```

index

المكان الذي فيه عنصر و يرجع (pos) التابع له

شرح: يبحث عن العنصر في List بطريقة (For)

وعندما يجد العنصر يرجع الـ index التابع له عن طريق (i.f)

```
return pos; → يرجع فيه نوع int
```

```
} //search
```

لأن (الفتكشة) معرف int

استخدمت فتكشة من نوع int وليست void لأنها ترجع فيه

ملاحظة

اضافة عنصر مكان العنصر الثاني (value)

void clist::add_element(int el,int value)

```
{
    if(!isfull()) → بتأكد اذا كانا غير طليعه
    {
        int pos=search(value);
        if(pos!=-1)
        {
            for(int i=length;i>pos;i--)
                list[i]=list[i-1];

            list[pos]=el;
            length++;
        }
        else
            cout<<"sorry value not found"<<endl;
    }
    else
        cout<<"sorry list is full"<<endl;
}
```

//add_element

void clist::delete_last()

```
{
    if(!isempty())
    {
        cout<<"delete last element:"<<list[length-1]<<endl;
        length--;
    }
    else
        cout<<"sorry list is empty"<<endl;
}
```

//delete_last

void clist::delete_first()

```
{
    if(!isempty())
    {
        cout<<"delete first element:"<<list[0]<<endl;
        for(int i=0;i<length-1;i++)
            list[i]=list[i+1];

        length--;
    }
    else
        cout<<"sorry list is empty"<<endl;
}
```

//delete_first

1

شرح :

يبحث عن (pos) التابع الـ value ويتأكد اذا كان موجوداً ام لا عن طريقه IF

وينقل كل العناصر من عند pos لتابع الـ value لنهاية الـ list خطوة للأمام .

وبصير الاشرير يوسر مع مكان pos

التابع value وتكون فارغة وبصير العنصر الجديد

حذف آخر قيمة في List

ليس ينقص طول الـ List (length--)

حذف اول قيمة في List

شرح :

ينقل كل العناصر خطوة للخلف يعني

(العنصر الثاني بصير مكان العنصر الاول)

وهكذا

وبصير ينقص طول الـ List لاننا حذفنا منه اول عنصر .

حذف العنصر حسب pos

شرح:

يتم حذف (pos) التابع (eL) وتأكد اذا كان موجود في صلاحيات Loop وبصية بيلش Loop
من (pos) ويجب كل العناصر التي بعده وبرمجة
خطوة للخلف وبقية من طول List
لنا صناعية EL

```
void clist::delete_element(int el)
{
    if(!isempty())
    {
        int pos=search(el);
        if(pos!=-1)
        {
            for(int i=pos;i<length-1;i++)
                list[i]=list[i+1];
            length--;
        }
        else
            cout<<"sorry element not found"<<endl;
    }
    else
        cout<<"sorry list is empty"<<endl;
}
```

delete_element

int clist::getfirst()

```
{
    return list[0];
}
```

getfirst

int clist::getlast()

```
{
    return list[length-1];
}
```

getlast

Main:

void main()

```
{
    clist l1;
    l1.addtofirst(50);
    l1.addtofirst(80);
    l1.addtofirst(40);
    l1.addtofirst(60);
    l1.addtofirst(11);
    l1.delete_less_than();
    l1.print();
}
```

11-stack:

Header:

```
const int size=5;
class cstack
{
public:
    cstack();
    bool isempty() const;
    bool isfull() const;
    void print() const;
    void push(int el);
    int search(int el);
    void pop();
    int gettop();
private:
    int stack[size];
    int top; → stack
```

Imp:

```
cstack::cstack()
{
    top=0;
}

//default constructor
bool cstack::isempty() const
{
    return top==0;
}

//isempty
bool cstack::isfull() const
{
    return top==size;
}

//isfull
void cstack::print() const
{
    if(!isempty())
    {
        for(int i=top-1;i>=0;i--)
            cout<<stack[i]<<" ";
        cout<<endl;
    }
    else
        cout<<"sorry stack is empty"<<endl;
}

//print
void cstack::push(int el)
{
    if(!isfull())
        stack[top++]=el;
    else
        cout<<"sorry stack is full"<<endl;
}

//push
```

بناءً إذا
stack فارغ

بناءً إذا
stack ممتلئ

→ For in stack

موجود

إضافة عنصر إلى stack

البحث عن العنصر بطريقة pos

نفسه شرح (list)

```
int cstack::search(int el)
{
    int pos=-1;
    if(!isempty())
    {
        for(int i=top-1;i>=0;i--)
            if(stack[i]==el)
            {
                pos=i;
                break;
            }
    }
    else
        cout<<"sorry stack is empty"<<endl;

    return pos;
}
```

//search

```
void cstack::pop()
{
    if(!isempty())
    {
        cout<<"delete top element:"<<stack[top-1]<<endl;
        top--;
    }
    else
        cout<<"sorry stack is empty"<<endl;
}
```

//pop

```
int cstack::gettop()
{
    return stack[top-1];
}
```

//gettop

```
Main:
void main()
{
```

```
    cstack s1;
    s1.push(11);
    s1.push(12);
    s1.push(13);
    s1.push(14);
    s1.push(15);
    s1.print();
}
```

حذف آخرية دهنه stack

حذف (Delete) في List
Last

يرجع آخرية دهنه stack
آخرية index

12-queue:

Header:

```
const int size=5;
class cQueue
{
public:
```

→ array حجم

```
cQueue();
bool isempty() const;
bool isfull() const;
void print() const;
void append(int el);
int search(int el);
void serve();
int retrieve();
private:
```

```
int Queue[size];
int count;
int front, rear;
```

→ Queue

توضيح:
 الحقل

توضيح:
 الحقل

```
Imp:
cQueue::cQueue()
{
```

```
count=0;
front=0;
rear=-1;
```

```
}//default constructor
bool cQueue::isempty() const
{
return count==0;
```

→ متأكد اذا Queue فارغ

```
}//isempty
bool cQueue::isfull() const
{
return count==size;
```

→ متأكد اذا Queue ممتلئ

```
}//isfull
void cQueue::print() const
{
```

```
if(!isempty())
```

```
{
for(int i=front; i=rear; i=(i+1)%size)
cout<<Queue[i]<<" ";
cout<<Queue[rear]<<endl;
```

→ For in Queue are

```
}
```

```
else
cout<<"sorry Queue is empty"<<endl;
```

```
}//print
```

شرح:
 1. نأخذ فيه front ونظن تلف

طول ما 1. لا تادي rear ولما اطلع من Loop لا تاربع اطلع rear
 size (i+1)
 نعمل على اعادة التفرقة من الحلقة

اضافة عنصر على Queue

شرح :-

```
void cQueue::append(int el)
{
```

```
    if(!isfull())
```

```
    {
        rear=(rear+1==size) ? 0 : rear+1;
```

```
        Queue[rear]=el;
        count++;
```

```
    }
    else
```

```
        cout<<"sorry Queue is full"<<endl;
```

```
}//append
```

```
int cQueue::search(int el)
{
```

```
    int pos=-1;
```

```
    if(!isempty())
```

```
    {
        for(int i=front; i!=rear; i=(i+1)%size)
```

```
        if(Queue[i]==el)
        {
```

```
            pos=i;
            break;
```

```
        }
    }
    if(pos==-1)
```

```
        if(Queue[rear]==el)
            pos=rear;
```

```
    }
    else
```

```
        cout<<"sorry Queue is empty"<<endl;
```

```
    return pos;
```

```
}//search
```

```
void cQueue::serve()
{
```

```
    if(!isempty())
    {
```

```
        cout<<"delete first element:"<<Queue[front]<<endl;
```

```
        front=(front+1==size) ? 0:front+1;
```

```
        count--;
```

```
    }
    else
```

```
        cout<<"sorry Queue is empty"<<endl;
```

```
}//serve
```

```
int cQueue::retrieve()
{
```

```
    return Queue[front];
```

```
}//retrieve
```

اذا كان $(rear + 1)$ يساوي

$(size)$ فنقل قيمته $rear$ لبادي صف

عنا $\sim$ اربع لبادي Queue

اما اذا كانت $(rear + 1)$ تساوي $(size)$

نزيد قيمته $rear$ واحد.

نبحث عن العنصر في Queue

نفس طرزم List

نحذف اول عنصر في Queue (front)

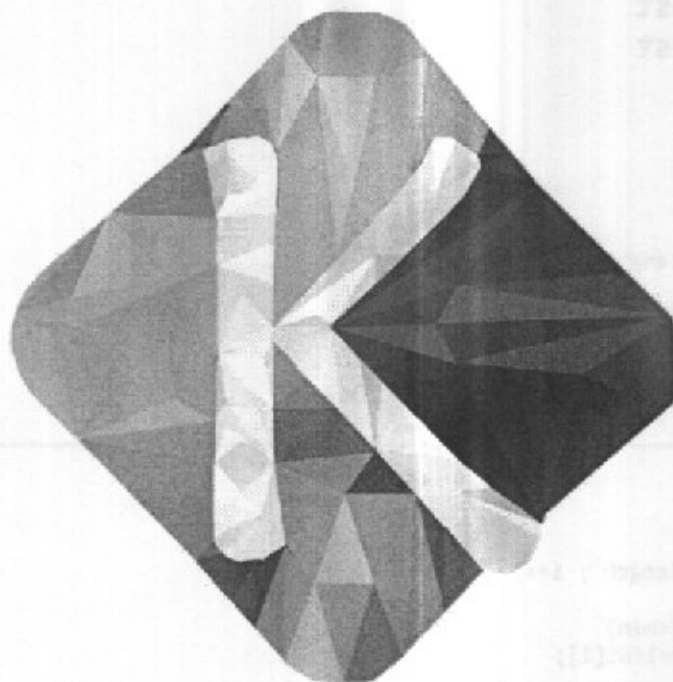
بغير قيمته ال front ونزيد قيمته الذي خلفه

نرجع قيمته للعنصر الاول.


```

Main:
void main()
{
    cQueue q1;
    q1.append(1);
    q1.append(2);
    q1.append(3);
    q1.append(4);
    q1.print();
}

```



KASIT TEAM

كودات دكتور نبيل العساف

كاسيت_تيم

اعداد الطالب عبد الكريم الحوراني



kasit team
كاسيت تيم

افكار عن list & stack & queue

Find max
find min
count even
count odd
delete last odd
delete last even
delete first even
delete first odd
delete second even
delete second odd
delete max
delete min
move first to last
move last to first

in list:

```
void clist::find_max()
{
    int max=list[0];
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]>max)
            max=list[i];
    }
    cout<<max<<endl;
}
```

```
void clist::find_min()
{
    int min=list[0];
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]<min)
            min=list[i];
    }
    cout<<min<<endl;
}
```

```

void clist::count_even()
{
    int count=0;
    for(int i=0 ; i<length ; i++)
        if(list[i]%2==0)
            count++;
    cout<<count<<endl;
}

```

```

void clist::count_odd()
{
    int count=0;
    for(int i=0 ; i<length ; i++)
        if(list[i]%2!=0)
            count++;
    cout<<count<<endl;
}

```

```

void clist::delete_last_odd()
{
    int odd;
    for(int i=0 ; i<length ; i++)
        if(list[i]%2!=0)
            odd=list[i];
    delete_element(odd);
    print();
}

```

```

void clist::delete_last_even()
{
    int even;
    for(int i=0 ; i<length ; i++)
        if(list[i]%2==0)
            even=list[i];
    delete_element(even);
    print();
}

```

```

void clist::delete_first_odd()
{
    int odd;
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]%2!=0)
        {
            odd=list[i];
            break;
        }
    }
    delete_element(odd);
    print();
}

```

```

void clist::delete_first_even()
{
    int even;
    for(int i=0 ; i<length ; i++)
        if(list[i]%2==0)
        {
            even=list[i];
            break;
        }
    delete_element(even);
    print();
}

void clist::delete_second_odd()
{
    int odd , count=0;
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]%2!=0)
        {
            odd=list[i];
            count++;
        }
        if(count==2)
        {
            delete_element(odd);
            print();
            break;
        }
    }
}

void clist::delete_second_even()
{
    int even , count=0;
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]%2==0)
        {
            even=list[i];
            count++;
        }
        if(count==2)
        {
            delete_element(even);
            print();
            break;
        }
    }
}

```

```

void clist::delete_max()
{
    int max=list[0];
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]>max)
            max=list[i];
    }
    delete_element(max);
    print();
}

void clist::delete_min()
{
    int min=list[0];
    for(int i=0 ; i<length ; i++)
    {
        if(list[i]<min)
            min=list[i];
    }
    delete_element(min);
    print();
}

void clist::move_first_to_last()
{
    int first=list[0];
    delete_first();
    addtolast(first);
}

void clist::move_last_to_first()
{
    int last= list[length-1];
    delete_last();
    addtofirst(last);
}

void clist::sort_list()
{
    clist l2;
    while(!isempty())
    {
        int min=list[0];
        for(int i=0 ; i<length ; i++)
            if(list[i]<min)
                min=list[i];
        l2.addtofirst(min);
        delete_element(min);
    }
    while(!l2.isempty())
    {
        addtofirst(l2.getfirst());
        l2.delete_first();
    }
    print();
}

```

In stack:

```
void cstack::find_max()
{
    cstack s2;
    int max=gettop();
    while(!isempty())
    {
        if(gettop()>max)
            max=gettop();
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
    cout<<max<<endl;
}
```

```
void cstack::find_min()
{
    cstack s2;
    int min=gettop();
    while(!isempty())
    {
        if(gettop()<min)
            min=gettop();
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
    cout<<min<<endl;
}
```

```
void cstack::count_even()
{
    cstack s2;
    int count=0;
    while(!isempty())
    {
        if(gettop()%2==0)
            count++;
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
    cout<<count<<endl;
}
```



```

void cstack::count_odd()
{
    cstack s2;
    int count=0;
    while(!isempty())
    {
        if(gettop()%2!=0)
            count++;
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
    cout<<count<<endl;
}

```

```

void cstack::delete_last_odd()
{
    cstack s2;
    int odd;
    while(!isempty())
    {
        if(gettop()%2!=0)
            odd=gettop();
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        if(s2.gettop()==odd)
        {
            s2.pop();
            continue;
        }
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::delete_last_even()
{
    cstack s2;
    int even;
    while(!isempty())
    {
        if(gettop()%2==0)
            even=gettop();
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        if(s2.gettop()==even)
        {
            s2.pop();
            continue;
        }
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::delete_first_even()
{
    cstack s2;
    int even;
    while(!isempty())
    {
        if(gettop()%2==0)
        {
            pop();
            break;
        }
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::delete_first_odd()
{
    cstack s2;
    int odd;
    while(!isempty())
    {
        if(gettop()%2!=0)
        {
            pop();
            break;
        }
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::delete_second_even()
{
    cstack s2;
    int even , count=0;
    while(!isempty())
    {
        if(gettop()%2==0)
        {
            count++;
            if(count==2)
            {
                pop();
                break;
            }
        }
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
}

```



```

void cstack::delete_second_odd()
{
    cstack s2;
    int odd ,count=0;
    while(!isempty())
    {
        if(gettop()%2!=0)
        {
            count++;
            if(count==2)
            {
                pop();
                break;
            }
        }
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::delete_max()
{
    cstack s2;
    int max=gettop();
    while(!isempty())
    {
        if(gettop()>max)
            max=gettop();
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        if(s2.gettop()==max)
        {
            s2.pop();
            continue;
        }
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::delete_min()
{
    cstack s2;
    int min=gettop();
    while(!isempty())
    {
        if(gettop()<min)
            min=gettop();
        s2.push(gettop());
        pop();
    }
    while(!s2.isempty())
    {
        if(s2.gettop()==min)
        {
            s2.pop();
            continue;
        }
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::move_first_to_last()
{
    cstack s2;
    int first=gettop();
    pop();
    while(!isempty())
    {
        s2.push(gettop());
        pop();
    }
    push(first);
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
}

```

```

void cstack::move_last_to_first()
{
    cstack s2;
    while(!isempty())
    {
        s2.push(gettop());
        pop();
    }
    int last = s2.gettop();
    s2.pop();
    while(!s2.isempty())
    {
        push(s2.gettop());
        s2.pop();
    }
    push(last);
}

```

```

Queue:
void cQueue::find_max()
{
    cQueue q2;
    int max=retrieve();
    while(!isempty())
    {
        if(retrieve() > max)
            max=retrieve();
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
    cout<<max<<endl;
}

```

```

void cQueue::find_min()
{
    cQueue q2;
    int min=retrieve();
    while(!isempty())
    {
        if(retrieve() < min)
            min=retrieve();
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
    cout<<min<<endl;
}

```

```

void cQueue::count_even()
{
    cQueue q2;
    int count=0;
    while(!isempty())
    {
        if(retrieve()%2==0)
            count++;
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
    cout<<count<<endl;
}

```

```

void cQueue::count_odd()
{
    cQueue q2;
    int count=0;
    while(!isempty())
    {
        if(retrieve()%2!=0)
            count++;
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
    cout<<count<<endl;
}

```

```

void cQueue::delete_last_odd()
{
    cQueue q2;
    int odd;
    while(!isempty())
    {
        if(retrieve()%2!=0)
            odd=retrieve();
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        if(q2.retrieve()==odd)
        {
            q2.serve();
            continue;
        }
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::delete_last_even()
{
    cQueue q2;
    int even;
    while(!isempty())
    {
        if(retrieve()%2==0)
            even=retrieve();
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        if(q2.retrieve()==even)
        {
            q2.serve();
            continue;
        }
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::delete_first_even()
{
    cQueue q2;
    int count=0;
    while(!isempty())
    {
        if(count==0)
            if(retrieve()%2==0)
            {
                serve();
                count++;
            }
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::delete_first_odd()
{
    cQueue q2;
    int count=0;
    while(!isempty())
    {
        if(count==0)
            if(retrieve()%2!=0)
            {
                serve();
                count++;
            }
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::delete_second_even()
{
    cQueue q2;
    int count=0;
    while(!isempty())
    {
        if(retrieve()%2==0)
        {
            count++;
            if(count==2)
            {
                serve();
                continue;
            }
        }
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
}

```



```

void cQueue::delete_second_odd()
{
    cQueue q2;
    int count=0;
    while(!isempty())
    {
        if(retrieve()%2!=0)
        {
            count++;
            if(count==2)
            {
                serve();
                continue;
            }
        }
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::delete_max()
{
    cQueue q2;
    int max=retrieve();
    while(!isempty())
    {
        if(retrieve() > max)
        {
            max=retrieve();
            q2.append(retrieve());
            serve();
        }
    }
    while(!q2.isempty())
    {
        if(q2.retrieve()==max)
        {
            q2.serve();
            continue;
        }
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::delete_min()
{
    cQueue q2;
    int min=retrieve();
    while(!isempty())
    {
        if(retrieve() < min)
            min=retrieve();
        q2.append(retrieve());
        serve();
    }
    while(!q2.isempty())
    {
        if(q2.retrieve()==min)
        {
            q2.serve();
            continue;
        }
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::move_first_to_last()
{
    cQueue q2;
    int first=retrieve();
    serve();
    while(!isempty())
    {
        q2.append(retrieve());
        serve();
    }
    q2.append(first);
    while(!q2.isempty())
    {
        append(q2.retrieve());
        q2.serve();
    }
}

```

```

void cQueue::move_last_to_first()
{
    cQueue q2;
    int last;
    while(!isempty())
    {
        q2.append(retrieve());
        last=retrieve();
        serve();
    }
    append(last);
    while(!q2.isempty())
    {
        if(q2.retrieve()==last)
        {
            q2.serve();
            continue;
        }
        append(q2.retrieve());
        q2.serve();
    }
}

```



```
void cQueue::move_last_to_first()
{
    cQueue q2;
    int last;
    while(!isempty())
    {
        q2.append(retrieve());
        last=retrieve();
        serve();
    }
    append(last);
    while(!q2.isempty())
    {
        if(q2.retrieve()==last)
        {
            q2.serve();
            continue;
        }
        append(q2.retrieve());
        q2.serve();
    }
}
```

File :

```
#include<fstream>
using namespace std;
void main()
{
    ifstream in ;
    ofstream out;

    in.open("input.txt");
    out.open("output.txt");

    int g1,g2;
    in>>g1>>g2;
    out<<g1<<" "<<g2<<endl;

    in.close();
    out.close();
}
```

stp_1

stp_2

stp_3

stp_4

stp_5

Bitwise :

```
#include<iostream>
using namespace std;
void main()
{
    cout<<(10 && 8)<<endl;
    cout<<(10 || 8)<<endl;
    cout<<(10 & 8)<<endl;
    cout<<(10 | 8)<<endl;
    cout<<(10 ^ 8)<<endl;
    cout<<(10 >> 2)<<endl;
    cout<<(10 << 2)<<endl;
    cout<<(~20)<<endl;
}
```

1
1
8
10
2
2
40
-21

Enumeration:

```
#include<iostream>
using namespace std;
void main()
{
    //enum seasan{spring , summer , autumn , winter };
    spring=0 , summer=1 , autumn=2 , winter=3
    enum seasan1{spring=10 , summer=15 , autumn ,
    winter=25};
    for(seasan1 i= spring ; i<= winter ; i=(seasan1)(i+1))
    {
        switch(i)
        {
            case 10:cout<<"spring"<<endl;break;
            case 15:cout<<"summer"<<endl;break;
            case 16:cout<<"autumn"<<endl;break;
            case 25:cout<<"winter"<<endl;break;
        }
    }
}
```

Output:

```
spring
summer
autumn
winter
```