```
/*
//-----------B站(bilibili.com)课程“从C到C++快速入门(2019版)”-------
//--------------https://www.bilibili.com/video/av40959422-----------
//--------------------源代码文件---------------
//--------------请关注：------------
//微博和B站：hw-dong
//网易云课堂：hwdong
//博客：https://a.hwdong.com
//腾讯课堂：http://hwdong.ke.qq.com
*/
#if 0
#define _CRT_SECURE_NO_WARNINGS
#include <cstdio>  //标准输入输出函数
#include <cmath>
#include <cstring> //字符串处理函数

int main() {
    printf("hello\n");
    double x = 3.14;
    printf("%lf %lf\n", sqrt(x), sin(x));

    char s[10] = "hello";
    puts(s);
    char s2[16];
    strcpy(s2, "world");
    puts(s2);
    strcat(s2, "sdfsdf");
    puts(s2);
    printf("%d %d\n", strlen(s), strlen(s2));
    return 0;
}
#endif

#if 0
#define _CRT_SECURE_NO_WARNINGS
#include <cstdio>  //标准输入输出函数
#include <cstring> //字符串处理函数
#include <malloc.h>
int main() {
# if 1
    char s[10];
    strcpy(s, "hello");
    puts(s);
#else
    char *s = (char *)malloc(12 * sizeof(char));
    strcpy(s, "hello world");
    puts(s);
#endif

}
#endif

#if 0

#include <iostream>  //C++标准输入输出流头文件
using namespace std;
```

```cpp
int main() {
    cout << "hello world!"<< endl;
    cout << "https://a.hwdong.com" << endl;
    cout << 3+4 << endl;
#if 0
    double radius;
    std::cin >> radius; //标准输入流对象cin 输入运算符>>
    cout << 3.14*radius*radius;
#endif
    std::cout << "  *\n";
    std::cout << " *  *\n";
    std::cout << " *  *\n";


    return 0;
}
#endif

#if 0
#include <iostream>
using namespace std;
void help() {
    cout << "=========简单计算器=========\n";
    cout << "请输入：左运算数  运算符  右运算符\n";
}

int main() {
    while (1) {
        help();
        double a, b;
        char op;
        cin >> a >> op >> b;
        if (op == '+')
            cout << a + b << endl;
        //...补充你的代码
    }
}
#endif

#if 0
#include <fstream>
#include <iostream>
#include <string>
using namespace std;
int main() {
    ofstream oF("test.txt");
    oF << 3.14 << " " << "hello world\n";
    oF.close();
    ifstream iF("test.txt");
    double d;
    string str;
    iF >> d >> str;
    cout<<d <<" "<< str<<endl;

    return 0;
}
```

```
113 #endif
114
115 #if 0
116 #include <iostream>
117 using namespace std;
118
119 int main() {
120     int a = 3, &r = a;
121     cout << a << '\t' << r << endl;
122     r = 5;
123     cout << a << '\t' << r << endl;
124     return 0;
125 }
126
127 #endif
128
129 #if 0
130 #include <iostream>
131 using namespace std;
132 void swap(int x, int y) {
133     cout << x << '\t' << y << endl;
134     int t = x;
135     x = y;
136     y = t;
137     cout << x << '\t' << y << endl;
138 }
139
140 int main() {
141     int a = 3, b = 4;
142     cout << a << '\t' << b << endl;
143     swap(a, b);
144     cout << a << '\t' << b  << endl;
145 }
146 #endif
147
148 #if 0
149 void swap(int *x, int *y) {
150     int t = *x;
151     *x = *y;
152     *y = t;
153 }
154 #include <iostream>
155 using namespace std;
156 int main() {
157     int a = 3, b = 4;
158     cout << a << '\t' << b << endl;
159     swap(&a, &b);
160     cout << a << '\t' << b << endl;
161 }
162 #endif
163
164 #if 0
165 void swap(int &x, int &y) {
166     int t = x;
167     x = y;
168     y = t;
```

```
169 }
170 #include <iostream>
171 using namespace std;
172 int main() {
173     int a = 3, b = 4;
174     cout << a << '\t' << b << endl;
175     swap(a, b);
176     cout << a << '\t' << b << endl;
177 }
178 #endif
179
180
181 #if 0
182 #include <iostream>
183 using namespace std;
184 void print(char ch, int n = 1) {
185     for (int i = 0; i < n; i++)
186         cout << ch;
187 }
188 int main() {
189     print('*'); cout << endl;
190     print('*',3); cout << endl;
191     print('*',5); cout << endl;
192 }
193 #endif
194
195 #if 0
196 #include <iostream>
197 using namespace std;
198 int add(int x,int y=2,int z=3) {
199     return x + y + z;
200 }
201 int main() {
202     cout << add(5)<<endl;
203     cout << add(5,7) << endl;
204     cout << add(5,7,9) << endl;
205 }
206 #endif
207
208 #if 0
209 #include <iostream>
210 using namespace std;
211 int add(int x, int y = 2) {
212     return x + y ;
213 }
214 double add(double x, double y = 2.0) {
215     return x + y;
216 }
217 int main() {
218     cout << add(5,3) << endl;
219     cout << add(5.3, 7.8) << endl;
220     cout << add((double)5, 7.8) << endl;//歧义性
221 }
222 #endif
223
224 #if 0
```

```cpp
#include <iostream>
using namespace std;
int add(int x, int y) {
    return x + y;
}
double add(double x, double y ) {
    return x + y;
}
int main() {
    cout << add(5, 3) << endl;
    cout << add(5.3, 7.8) << endl;
//  cout << add("hello", "world") << endl;
}
#endif

#if 0
#include <iostream>
#include <string>
using namespace std;

template<typename T>
T add(T x, T y) {
    return x + y;
}
int main() {
#if 0
    cout << add<int>(5, 3) << endl;
    cout << add<double>(5.3, 7.8) << endl;
    cout << add<int>(4, 6) << endl;
    cout << add<string>("hello", "world") << endl;
#else
    cout << add(5, 3) << endl;
    cout << add(5.3, 7.8) << endl;
    cout << add((double)5, 7.8) << endl; //歧义性
#endif
}
#endif


#if 0
#include <iostream>
#include <string>
using namespace std;
int main() {
    string s = "hello", s2("world");
    //访问运算符.
    cout << s.size() << endl;
    string s3 = s.substr(1, 3);
    cout << s3<< endl;

    string s4 = s + " " + s2;
    cout << s4 << endl; //"hello world"

    s4[0] = 'H';
    s4[6] = 'X';
    cout << s4 << endl;
```

```
281
282     int pos = s4.find("orl");
283     cout << pos << endl;
284     s4.insert(3, "ABCDE");
285     cout << s4 << endl;
286
287     for (int i = 0; i < s4.size(); i++)
288         cout << s4[i] << "-";
289     cout << "\n";
290
291 }
292 #endif
293
294 #if 0
295 #include <iostream>
296 using std::cout;
297 int main() {
298     int arr[] = { 10,20,30,40 }; //大小固定，以后不能添加更多int值
299     for (int i = 0; i < 4; i++)
300         cout << arr[i] << '\t';
301     cout << '\n';
302 }
303 #endif
304
305 #if 0
306 #include <iostream>
307 #include <vector>
308 using namespace std;
309 int main() {
310     vector<int> v = { 7, 5, 16, 8 };
311     //push_back()，最后添加一个元素
312     v.push_back(25);
313     v.push_back(13);
314
315     //成员函数size()、下标运算符[]
316     for (int i = 0; i < v.size(); i++)
317         cout << v[i] << '\t';
318     cout << '\n';
319
320     v.pop_back();
321     for (int i = 0; i < v.size(); i++)
322         cout << v[i] << '\t';
323     cout << '\n';
324
325     v.resize(2);
326
327     for (int i = 0; i < v.size(); i++)
328         cout << v[i] << '\t';
329     cout << '\n';
330 }
331 #endif
332
333 #if 0
334 /*
335 指针就是地址，变量的指针就是变量的地址。
336 指针变量就是存储指针（地址）的变量。
```

```
337 */
338 #include <iostream>
339 using namespace std;
340 int main() {
341     int a=3;
342     int *p = &a;   //取地址运算符&用于获得a的地址：&a
343     cout << p << '\t' << &a << endl;
344     //取内容运算符*用于获得指针指向的变量(内存块)
345     cout << *p << '\t' << a << endl;     //*p就是a
346     *p = 5;                              //即a = 5;
347     cout << *p << '\t' << a << endl;
348 #if 1
349     int *q = p;             //q和p值相同，都是a的地址(指针)
350     cout << *p << '\t' << *q << '\t' << a << endl;
351     char *s = &a; //int *
352 #endif
353 }
354 #endif
355
356 #if 0
357 /*
358 用指针访问数组元素
359 */
360 #include <iostream>
361 using namespace std;
362 int main() {
363     int arr[] = { 10,20,30,40 };
364     int *p = arr;//数组名就是数组第一个元素的地址，即arr等于&(arr[0])
365     // p[i]就是*(p+i)
366     cout << *(p + 2) << '\t' << p[2] << '\t' << arr[2] << endl;
367
368     for (int *q = p + 4; p < q; p++)
369         cout << *p << '\t';
370     cout << '\n';
371 }
372 #endif
373
374 #if 0
375 /*
376 malloc free realloc
377   动态内存分配：new用于申请内存块、delete用于释放内存块
378   T *p = new T;
379   delete p;
380   T *q  = new T[5];
381   delete[] q;
382 */
383 #if 0
384 // 堆存储区
385 #include <iostream>
386 using namespace std;
387 int main() {
388     int *p = new int; //malloc
389     *p = 3;
390     cout << p << '\t' << *p << endl;
391     delete p;   //内存泄漏
392     p = new int;
```

```
393     *p = 5;
394     cout << p << '\t' << *p << endl;
395     delete p;
396 }
397 #else
398 #include <iostream>
399 using namespace std;
400 int main() {
401     int n = 4;
402     int *p = new int[n];
403     for (int i = 0; i < n; i++)
404         p[i] = 2 * i + 1;
405 
406     for (int *q = p + n; p < q; p++)
407         cout << *p << '\t';
408     cout << '\n';
409 
410     char *s = (char *)p;
411     char ch = 'A';
412     int n2 = n * sizeof(int) / sizeof(char);
413     for (int i = 0; i < n2; i++)
414         s[i] = ch + i;
415 
416     for (char *r = s+n2; s < r; s++)
417         cout << *s;
418     cout << '\n';
419 
420     delete[] p;
421 }
422 #endif
423 #endif
424 
425 #if 0
426 /*
427 输入一组学生成绩(姓名和分数)，输出：平均成绩、最高分和最低分。
428 当然，也要能输出所有学生信息
429 
430 */
431 #include <iostream>
432 #include <string>
433 #include <vector>
434 using namespace std;
435 struct student{
436     string name;
437     double score;
438     void print();
439 };
440 void student::print() {
441     cout << name << " " << score << endl;
442 }
443 
444 int main() {
445 #if 0
446     student stu;
447     stu.name = "Li Ping";
448     stu.score = 78.5;
```

```
449     stu.print();
450 #endif
451     vector<student> students;
452 
453     while (1) {
454         student stu;
455         cout << "请输入姓名 分数:\n";
456         cin >> stu.name >> stu.score;
457         if (stu.score < 0) break;
458         students.push_back(stu);
459     }
460     for (int i = 0; i < students.size(); i++)
461         students[i].print();
462 
463     double min = 100, max=0, average = 0;
464     for (int i = 0; i < students.size(); i++) {
465         if (students[i].score < min) min = students[i].score;
466         if (students[i].score > max) max = students[i].score;
467         average += students[i].score;
468     }
469     average /= students.size();
470     cout << "平均分、最高分、最低分："
471         << average << " " << max << " " << min << endl;
472 
473 }
474 #endif
475 
476 #if 0
477 /*
478     this指针: 成员函数实际上隐含一个this指针。
479 */
480 #include <iostream>
481 #include <string>
482 using namespace std;
483 
484 struct student {
485     string name;
486     double score;
487     void print() {
488         cout << this->name << " " << this->score << endl;
489     }
490 };
491 int main() {
492     student stu;
493     stu.name = "Li Ping";
494     stu.score = 78.5;
495     stu.print();   // print(&stu);
496 }
497 #endif
498 
499 
500 
501 #if 0
502 /*
503     struct和class区别:
504     struct里的成员默认是public(公开的)
```

```
505     class里的成员默认是private(私有的)
506 */
507 #include <iostream>
508 #include <string>
509 using namespace std;
510
511 class student{
512 public: //接口
513     void print() {
514         cout << this->name << " " << this->score << endl;
515     }
516     string get_name() { return name; }
517     double get_score() { return score; }
518     void set_name(string n) { name = n; }
519     void set_score(double s) { score = s; }
520 private:
521     string name;
522     double score;
523 };
524 int main() {
525     student stu;
526
527 //  stu.name = "Li Ping";
528 //  stu.score = 78.5;
529     stu.set_name("Li Ping");
530     stu.set_score(78.5);
531     stu.print();   // print(&stu);
532     cout << stu.get_name() << " " << stu.get_score() << endl;
533 }
534 #endif
535
536 #if 0
537 /*
538 构造函数:  函数名和类名相同且无返回类型的成员函数。
539 */
540
541 #include <iostream>
542 #include <string>
543 using namespace std;
544
545 class student{
546     string name;
547     double score;
548 public:
549     student(string n,double s){ //不是默认构造函数
550         name = n; score = s;
551         cout << "构造函数\n";
552     }
553     void print() {
554         cout << this->name << " " << this->score << endl;
555     }
556 };
557 int main() {
558     student stu("LiPing",80.5); //在创建一个类对象时会自动调用称为“构造函数”的成员函
      数
559     stu.print();
```

```
560     student students[3];
561
562 }
563
564 #endif
565
566 #if 0
567 /* 运算符重载：针对用户定义类型重新定义运算符函数
568 */
569 #include <iostream>
570 #include <string>
571 using namespace std;
572 class student {
573     string name;
574     double score;
575 public:
576     student(string n, double s) {
577         name = n; score = s;
578     }
579     //友元函数
580     friend ostream& operator<<(ostream &o, student s);
581     friend istream& operator>>(istream &in, student &s);
582 };
583
584 ostream& operator<<(ostream &o, student s) {
585     cout << s.name << "," << s.score << endl;
586     return o;
587 }
588 istream& operator>>(istream &in, student &s) {
589     in >> s.name >> s.score;
590     return in;
591 }
592
593 int main() {
594     student stu("LiPing", 80.5);
595     cin >> stu; //operator>>(cin,stu)
596     cout << stu; //operator<<(cout,stu)
597 }
598
599 #endif
600
601 #if 0
602 #include <iostream>
603 #include <string>
604 using namespace std;
605
606 class Point{
607     double x, y;
608 public:
609     double operator[](int i) const{ //const函数
610         if (i == 0) return x;
611         else if (i == 1) return y;
612         else throw "下标非法!"; //抛出异常
613     }
614     double& operator[](int i) {
615         if (i == 0) return x;
```

```
616         else if (i == 1) return y;
617         else throw "下标非法!"; //抛出异常
618     }
619     Point(double x_,double y_) {
620         x = x_; y = y_;
621     }
622     Point operator+(const Point q) {
623         return Point(this->x+q[0],this->y + q[1]);
624     }
625
626     //友元函数
627     friend ostream & operator<<(ostream &o, Point p);
628     friend istream & operator>>(istream &i, Point &p);
629 };
630
631 ostream & operator<<(ostream &o, Point p) {
632     o <<p.x << " " << p.y<< endl;
633     return o;
634 }
635 istream & operator>>(istream &i, Point &p) {
636     i >> p.x >> p.y;
637     return i;
638 }
639 #if 0
640 Point operator+(const Point p,const Point q) {
641     return Point(p[0] + q[0], p[1] + q[1]);
642 }
643 #endif
644
645 int main() {
646     Point p(3.5, 4.8),q(2.0,3.0);
647 #if 0
648 //  cin >> p;
649     cout << p;
650     cout << p[0] << "-" << p[1] << endl; //p.operator[](0)
651     p[0] = 3.45; p[1] = 5.67;
652     cout << p;
653 #endif
654     cout << p<<q;
655     Point s = p + q; //p.operator+(q)  vs  operator+(p,q)
656     cout << s;
657 }
658 #endif
659
660 #if 0
661 #include <iostream>
662
663 using namespace std;
664
665 class String {
666     char *data; //C风格的字符串
667     int n;
668 public:
669     ~String() {
670         cout <<n<< " 析构函数!\n";
671         if(data)
```

```
672             delete[] data;
673     }
674 #if 1
675     String(const String &s) { //硬拷贝
676         cout << "拷贝构造函数!\n";
677         data = new char[s.n + 1];
678         n = s.n;
679         for (int i = 0; i < n; i++)
680             data[i] = s.data[i];
681         data[n] = '\0';
682     }
683 #endif
684     String(const char *s=0) {
685         cout << "构造函数!\n";
686         if (s == 0) {
687             cout << "s==0\n";
688             data = 0; n = 0; return;
689         }
690         const char *p = s;
691         while (*p) p++;
692         n = p - s;
693         data = new char[n + 1];
694         for (int i = 0; i < n; i++)
695             data[i] = s[i];
696         data[n] = '\0';
697     }
698     int size() { return n; }
699     char operator[](int i)const {
700         if (i<0 || i>=n ) throw "下标非法";
701         return data[i];
702     }
703     char& operator[](int i) {
704         if (i < 0 || i >= n) throw "下标非法";
705         return data[i];
706     }
707 };
708
709 ostream & operator<<(ostream &o, String s) {
710     for (int i = 0; i < s.size(); i++)
711         cout << s[i];
712     return o;
713 }
714 void f() {
715     String str,str2("hello world");
716     str2[1] = 'E';
717 //  cout << str2 << endl;
718
719 #if 1
720     String s3 = str2; //拷贝构造函数
721     cout << s3 << endl;
722     s3[3] = 'L';
723     cout << s3 << endl;
724     cout << str2 << endl;
725 #endif
726
727 }
```

```
728 int main() {
729     f();
730 }
731 #endif
732
733 #if 1
734 /*类
735    模拟vector<int>的类Vector
736 */
737 #include <iostream>
738 #include <string>
739 using namespace std;
740
741 class student {
742     string name;
743     double score;
744 public:
745     student(string n="no", double s=0) {
746         name = n; score = s;
747     }
748     friend ostream& operator<<(ostream &o, student s);
749 };
750
751 ostream& operator<<(ostream &o, student s) {
752     cout << s.name << "," << s.score << endl;
753     return o;
754 }
755
756 //类模板
757 template<typename T>
758 class Vector {
759     T *data;
760     int capacity;
761     int n;
762 public:
763     Vector(int cap=3) {
764         data = new T[cap];
765         if (data == 0) {
766             cap = 0; n = 0;
767             return;
768         }
769         capacity = cap;
770         n = 0;
771     }
772     void push_back(T e) {
773         if (n == capacity) {//空间已经满
774             cout << "增加容量! \n";
775             T *p = new T[2 * capacity];
776             if (p) {
777                 for (int i = 0; i < n; i++)
778                     p[i] = data[i];
779                 delete[] data;
780                 data = p;
781                 capacity = 2*capacity;
782             }
783             else {
```

```
784                 return;
785             }
786         }
787         data[n] = e;
788         n++;
789     }
790     T operator[](int i) const{
791         if (i < 0 || i >= n) throw "下标非法!";
792         return data[i];
793     }
794     int size() {
795         return n;
796     }
797 };
798 int main() {
799     Vector<student> v;
800     v.push_back(student("Li",45.7));
801     v.push_back(student("Wang", 45.7));
802     v.push_back(student("zhao", 45.7));
803 
804     for (int i = 0; i < v.size(); i++)
805         cout << v[i] ;
806     cout << endl;
807 
808     v.push_back(student("zhang", 45.7));
809     v.push_back(student("Liu", 45.7));
810     for (int i = 0; i < v.size(); i++)
811         cout << v[i];
812     cout << endl;
813 
814 #if 0
815 #if 1
816     Vector<int> v;
817     v.push_back(3);
818     v.push_back(4);
819     v.push_back(5);
820 
821     for(int  i = 0 ; i<v.size();i++)
822         cout<<v[i]<<'\t';
823     cout << endl;
824 
825     v.push_back(6);
826     v.push_back(7);
827     for (int i = 0; i < v.size(); i++)
828         cout << v[i] << '\t';
829     cout << endl;
830 #else
831     Vector<string> v;
832     v.push_back("hello");
833     v.push_back("world");
834     v.push_back("sdfasdf");
835 
836     for (int i = 0; i < v.size(); i++)
837         cout << v[i] << '\t';
838     cout << endl;
839 
```

```
840         v.push_back("ggg");
841         v.push_back("hhh");
842         for (int i = 0; i < v.size(); i++)
843             cout << v[i] << '\t';
844         cout << endl;
845     #endif
846     #endif
847
848     }
849     #endif
```