码图中常见问题解决方案

一、程序得分错误的排查方式

同学们所提交的程序会有两种得分错误：编译错误和运行测试错误。下面我们给出这两种错误的排查方案。

1、编译错误

针对程序的编译错误，可以按照一下步骤排查错误（案例一）：

1	hello world	C++	2018-11-08 13:15:14.0	编译错误	0	查看 | 下载

图 1

如图一所示，同学可以在作业状态中查看此次提交的编译错误。点击查看可以浏览具体的编译错误信息，图二中展示一个程序编译错误的细节信息。

成绩详单	
扣分:100	编译错误 C:\aptat\test_dir\assignment\S\P\hello world_2013060201006_20181108131514\hello world_2013060201006_20181108131514.cpp(1) : fatal error C1083: 无法打开包括文件: "stdoi.h" : No such file or directory

图二

码图系统目前使用的编译器为 VS2008 版本，该版本的 C++语言不支持 C11 语法，如智能指针，auto 等。

2、 运行测试错误

在编译成功后，系统将对程序进行测试。当作业状态为 score 时，代表测试成功。同学们可以通过图三作业状态中的查看按钮查看图四的扣分信息。

5	2_实现指定函数swap	C++	2017-06-09 14:09:52.0	score	80	查看 | 下载

图三

成绩详单	
扣分:10	When test group(6) step(1),the input is (6),time out.
扣分:10	When test group(5) step(1),the input is (5),the real output is not right.

图四

扣分信息目前分为两种：time_out 和 output is not right。下面针对这两种扣分信息进行解释。

①time_out。码图系统对被测试程序具有时间限制。如果被测试程序在规定的时间内没有进行输出，那么将会得到 time_out 错误。time_out 的错误可能由以下两种原因：被测试程序算法时间复杂度达不到要求，需要对算法进行优化；被测试程序的 input 数据与期待的 input 数据不符合，导致被测试程序阻塞等待 io 输入（例如教师作业中要求从标准输入中获取三个字符，但被测试程序调用了四个 getchar()函数，第四个 getchar()函数获取不到输入字符导致程序运行超时）。

② output is not right。被测试程序的输出与期望输出不一致。该错误的产生原因为代码错误。比如输出时，标点符号全角或半角无法匹配（案例二）；没有考虑边界条件等。

此外，有些同学会出现本地机器上程序手动测试正确，上传到码图却发现测试不通过的现象。这种现象可以通过以下思路解决：

①输出中全角符号与半角符号是否混淆，是否与要求的输出完全一致。例如程序要求输出 letters:5,digits:5，而有位同学却输出 letters:5,digits=5。这种错误往往很难发现，一定

要仔细检查。

②debug 模式与 release 模式下程序表现不一致（案例四）。很多程序在 debug 模式下，即使代码有数组越界问题，可能也会正常执行。而码图系统使用 release 模式进行编译。

③编译器的不同导致测试结果不同（案例五）。局部变量未初始化在不同编译器下表现不同，有且编译器会帮你默认初始化，有些编译器不会处理，有些编译器会随机初始化码图使用的编译器为 Vs2008。

④混合编译题型的陷阱（案例三）。混合编译题型没有提供 main 函数，同学们尝试写 main 函数进行测试时，一定要注意：系统的 main 函数可能会把输入进行修改后传给你实现的函数。因此当出现扣分，扣分细则里面的输入数据可能与测试实际传入函数内的数据不一致。

因此，同学们可以通过使用 VS2008 以 Release 模式编译运行程序，便可让码图系统的测试与本机系统的测试保持一致性。

如果以上排查方式均不能解决问题，请向老师协助排查问题。本文档在结尾提供一些程序错误测试的案例，供大家参考。

二、码图无法访问的排查方式

码图的网址为 http://222.197.179.3:1024/aptat。当该网站无法访问时，可以通过以下步骤进行排查：手机使用移动网络访问本网站，如果手机端可以访问，说明是电脑端网络出现问题。如果手机端同样无法访问，请尽快通知老师。

附录 程序错误案例

案例一　编译错误的处理方法

码图中第 108 题，提交一下代码发现编译错误，如图 7。

```
# include<stdio.h>
void * reversememcpy
(void * destination,void *
source,int num)
{
    int i;
    char *p,*q;
    p=(char *)destination;
    q=(char *)source;
    if (num>strlen(q))
        printf("error");
    else
    {
        for (i=num-1;i>=0;i--)
        {
            p[num-1-i]=q[i];
        }
        p[num]='\0';
    }
}

int main()
{
char ch[100],a[100];
int n;
gets(ch);
scanf("%d",&n);
reversememcpy(a,ch,n);
printf("%s",a);
}
```

作业状态						
题目编号	名称	提交语言	时间	状态	分数	详情
108	4.5 字符串逆序输出	C++	2018-11-08 17:27:02.0	编译错误	0	查看 | 下载

图七

点击查看按钮，可以查看相关编译错误，如图八。

成绩详单	
扣分:100	编译错误 C:\aptat\test_dir\assignment\S\P\4.5 字符串逆序输出_2013060201006_20181108172702\4.5 字符串逆序输出_2013060201006_20181108172702.cpp(9) : error C3861: “strlen”：找不到标识符

图八

可以看到编译错误是由于找不到 strlen 标识符。因此可以推断是由于没有包含 strlen 相关的头文件导致的。在百度上搜索 strlen 头文件，将对应的头文件#include <string.h>添加到代码中，继续提交即可。

本系统中使用的头文件为 VS2008 版本的头文件，对其他版本的头文件不予以支持。由于 Vs 的升级，很多函数的头文件发生了变迁，可能在自己机器上不会出现编译错误，但是在码图中会出现编译错误。希望大家能够注意到此问题。

案例二　调换 if 和 else 顺序却导致程序出错?

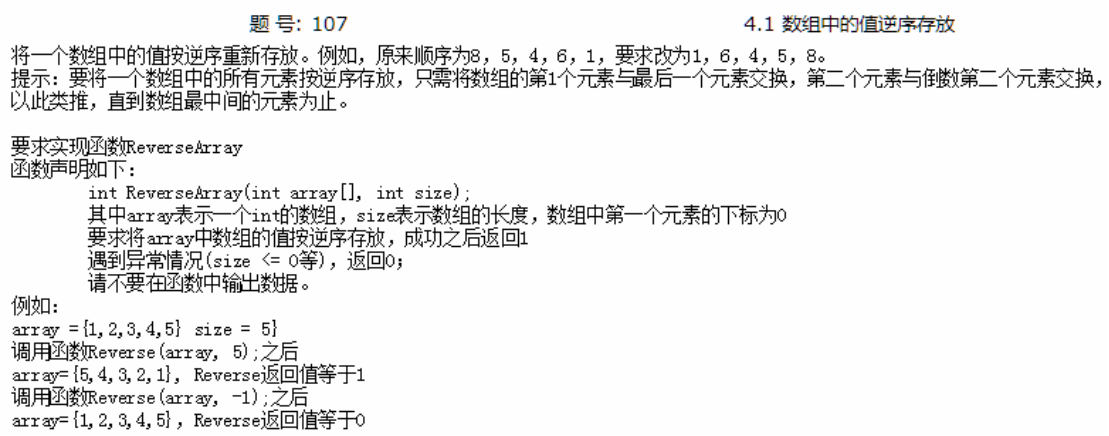
有位同学反映同样的程序，调换了 if 和 else if 的顺序后程序测试结果不一致。下面展示相关代码。

```
#include<stdio.h>
#include<string.h>
int main(){
    int a=0,b=0,i,d;
    char c[50];
    gets(c);
    d=strlen(c);
    for(i=0;i<d;i++){
      if(c[i]>='0'&&c[i]<='9')
        a++;
      else if((c[i]>='a'&&c[i]<='z')
      ||(c[i]>='A'&&c[i]<='Z'))
        b++;
    }
    printf("letters:%d,digits:%d",b,a);
}
```

```
#include<stdio.h>
#include<string.h>
int main(){
    char c[50];
    int a=0,b=0,i,d;
    gets(c);
    d=strlen(c);
    for(i=0;i<d;i++){
      if((c[i]>='a'&&c[i]<='z')||
      (c[i]>='A'&&c[i]<='Z'))
        a++;
      else if(c[i]>='0'&&c[i]<='9')
        b++;
    }
    printf("letters:%d,digits=%d",a,b);
}
```

仔细对比程序逻辑，发现确实只调换了 if 和 else if 语句，两个程序的结果应该是相同搞得。经过实际测试发现，左边的程序得分 100 分，而右边的程序却只有 0 分。管理员经过调试后发现，右侧程序的错误原因在于输出与期望输出不匹配。因为右侧程序输出将冒号修改为了等号。

案例三　混合编译题型的陷阱



题 号：107　　4.1 数组中的值逆序存放

将一个数组中的值按逆序重新存放。例如，原来顺序为8，5，4，6，1，要求改为1，6，4，5，8。
提示：要将一个数组中的所有元素按逆序存放，只需将数组的第1个元素与最后一个元素交换，第二个元素与倒数第二个元素交换，以此类推，直到数组最中间的元素为止。

要求实现函数ReverseArray
函数声明如下：

```
int ReverseArray(int array[], int size);
```

其中array表示一个int的数组，size表示数组的长度，数组中第一个元素的下标为0
要求将array中数组的值按逆序存放，成功之后返回1
遇到异常情况(size <= 0等)，返回0；
请不要在函数中输出数据。

例如：
array ={1,2,3,4,5} size = 5}
调用函数Reverse(array, 5);之后
array={5,4,3,2,1}, Reverse返回值等于1
调用函数Reverse(array, -1);之后
array={1,2,3,4,5}, Reverse返回值等于0

同学提交程序后出现以下错误：扣分:10　When test group(9) step(1),the input is (8 6 8 9 8 7 2 4 5 1 2),time out.扣分:5　When test group(10) step(1),the input is (8 8 8 9 8 7 2 4 5 1 2),time out.

该程序为混合编译题型，同学只需实现函数 ReverseArray。于是同学们自己写一个主函数进行测试，数据完全按照扣分信息中的 input 进行输入，发现函数完全正确，不会出现异常。

导致此问题的原因在于，在系统的 main 函数中，获取到 input 输入之后对输入数据做了处理。本题目的测试点 9 和 10 中，虽然获取到数组，但是并没有将该数组传递给 ReverseArray 函数，而是将 null 作为数组参数传给该函数。由于同学们的函数中没有异常处理，对野指针进行操作，导致程序出错。

案例四　编译器的 Debug 模式和 Release 模式的不同

题 号：109　　4.6 统计一行字符的单词数

输入一行字符，统计其中包括多少单词，单词之间用空格分隔。

输入输出格式要求：
　　输入格式：a b c d回车（空格可在任意位置，列如行头行尾可能会有空格。另外数字和字母一样，也可作为单词）
　　输出格式：单词数
　　输入的字符串长度最长为256个字符
例如：
输入：（空格）a a 112 c（空格）回车
输出：4

针对此程序，有位同学写了一下代码：

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
	char ch[257];
	int i, count=0;
	gets_s(ch, 256);
	for (i = 0; i < 256; i++)
	{
		if (ch[i] != ' '&&(ch[i + 1] == ' '||ch[i+1]=='\0'))
			count++;
	}
	printf("%d", count);
	system("pause");
	return 0;
}
```

该同学在本机对此程序进行测试，发现程序完全正确。在码图中提交代码，发现程序得分为 0。他查看具体测试信息，发现以下错误。

成绩详单	
扣分:10	When test group(6) step(1),the input is (asfjdsjaf sdfjidsjf dsfjidsjf dfjjfdjsafj ejj),the real output is not right.
扣分:20	When test group(3) step(1),the input is (aj ije wijij),the real output is not right.
扣分:10	When test group(5) step(1),the input is (dsfjidsjf jijiejjf dsafjij jdsaifj ewj),the real output is not right.
扣分:20	When test group(4) step(1),the input is (jidsafj jjje jijdsf jiwj jidsa),the real output is not right.
扣分:20	When test group(1) step(1),the input is (a a b c dew fjie),the real output is not right.
扣分:20	When test group(2) step(1),the input is (a e j ejj bgjg aaj 34),the real output is not right.

他将 input 数据记录下来，回到本机测试，发现程序完全正确。

这个程序确实存在问题，而且这个问题隐藏的很深刻，恰巧在码图系统中可以暴露。那就

是该程序存在数组越界的情况，他使用 ch[257]数组存储字符串，在对字符串进行处理时，却遍历全数组。因为他本机程序编译模式为 debug 模式，debug 模式中将 char 数组中没有使用的 char 全部置为 0xfe，在这种情况下程序执行正确。但是码图系统编译模式为 release 模式，在该模式下，编译器会将 char 数组进行随机初始化。在运行中，char 数组除了字符串结尾有'\0'外，剩余随机初始化部分还有'\0'，导致重复计数产生错误。

案例五　编译器的不同导致程序测试出错

有同学曾向我反映，45 题约瑟夫游戏中，本机程序执行完全正确，但是无法在系统中通过测试。由于代码过长，下面只选取问题代码展示：

```
int deleteElements(ListNodePtr head, int n, int m,int k){
    int out=n-k;
    int count_out;
    ListNodePtr p=head;
    ListNodePtr s;
    while(count_out<out)
    {
        for(int i=1;i<m;i++)
        {
            s=p;
            p=p->next;

        }

        if(p==head){
            head=head->next;
        }
        s->next=p->next;
        count_out++;
        p=s->next;
    }
    return 0;
}
```

该代码有一个明显的问题，那就是 count_out 没有初始化。不同编译器对未初始化的处理不同。他使用的 DEV C 编译器，可以通过运行。但是在高版本 VS 下，程序会报错。