華中科技大學

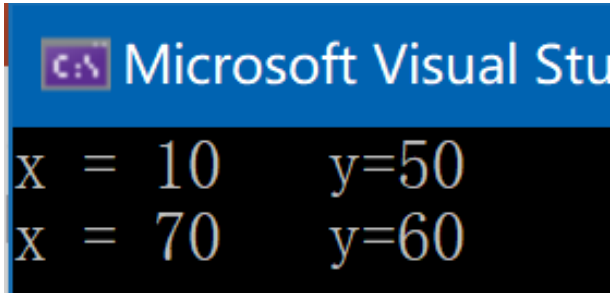
基本C语言知识小测验

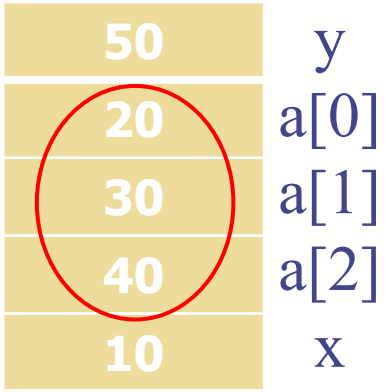
许向阳

xuxy@hust.edu.cn

小测验——数组访问

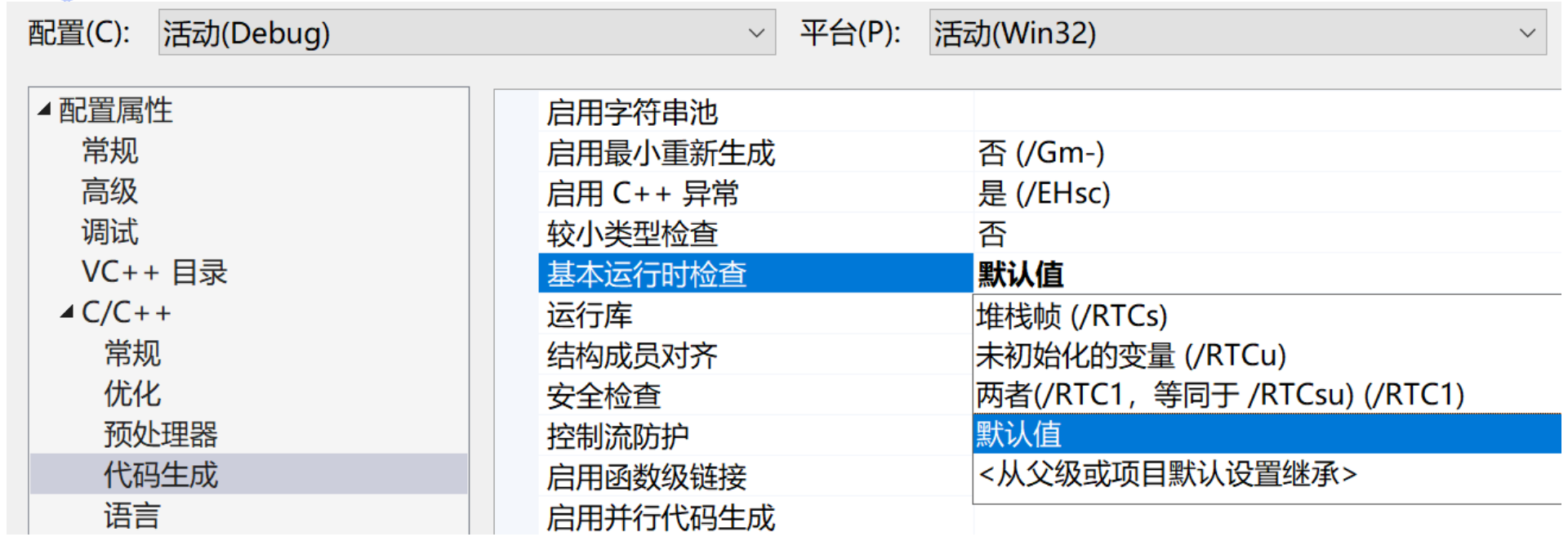
```
#include <iostream>
int main()
{
    int  x=10;
    int  a[3] = { 20,30,40 };
    int  y = 50;
    printf("x = %d   y=%d\n", x, y);
    a[-1] = 60;
    a[3] = 70;
    printf("x = %d   y=%d\n", x, y);
    return 0;
}
```





Q：运行结果是什么？ 为什么？

常见错误 数组越界

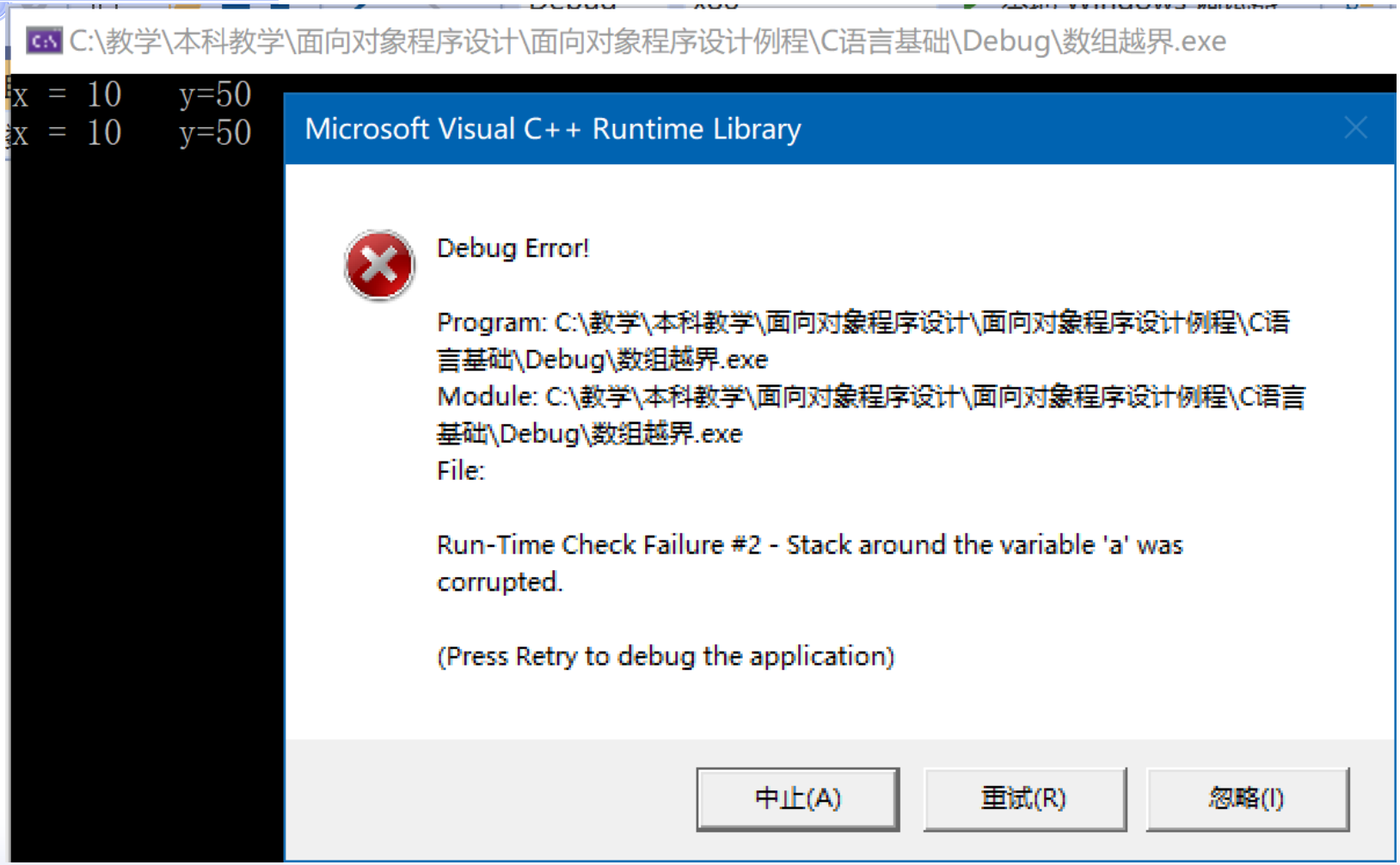


项目属性：C/C++ → 代码生成 → 基本运行时检查，设置为默认值，变量空间分配是紧凑的；才会出现上面的结果。

常见错误 数组越界

项目属性：C/C++ → 代码生成 → 基本运行时检查，
设置为两者，
变量空间分配非紧凑的，即变量不相邻，
会出现异常对话框。

常见错误 数组越界



Stack around the variable ‘a’ was corrupted.

常见错误 数组越界

基本运行时检查

两者: 堆栈帧、未初始化的变量

思考题：

- 何时进行越界检查？
- 如何进行越界检查？
- 能否发生了越界，而系统又未检测出来？
- 在没有越界检查时，数组越界 会 or 不会导致程序崩溃？

常见错误 数组越界

基本运行时检查

两者: 堆栈帧、未初始化的变量

思考题：

- 难道在编译时不检查？

```
int  x,y;
y=x;    // 编译报错：使用了未初始化的局部变量 x
```

- 有无在编译时检查不出来是否初始化的情况？

```
if (…)     x=10;
y=x;
```

- 如何在运行时，检查到有未初始化的变量？

常见错误 数组越界

基本运行时检查

两者: 堆栈帧、未初始化的变量

- 运行时，检查到有未初始化的变量

```
int x, y, z=10;
if (z == 1)  x = 10;
y = x;
static
int  j
cout <<
;++;
```

已引发异常

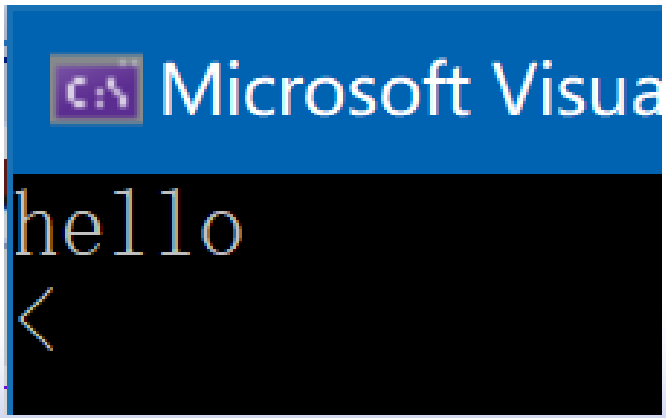
Run-Time Check Failure #3 - The variable 'x' is being used without being initialized.

复制详细信息 | 启动 Live Share 会话...

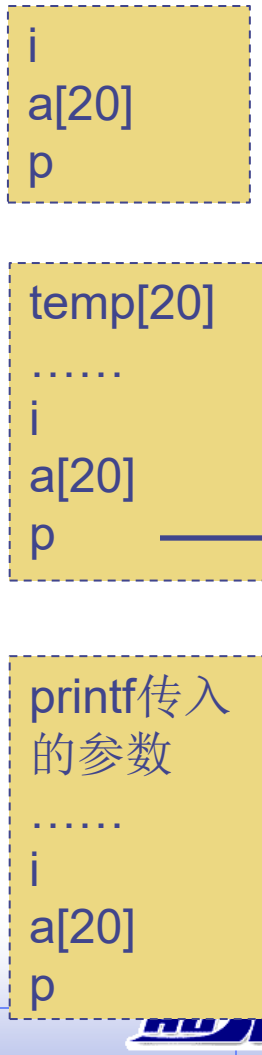
设置一个“影子变量”，初值为0，用于表示对应单元未初始化。若执行了给变量的赋值语句，则会将“影子变量”置为1. 在使用变量时，有判断影子变量是否为1的语句。

小测验——函数返回局部地址

```
#include <iostream>
char* f()
{
    char temp[20];
    strcpy_s(temp, "hello");
    return temp;
}
```




```
int main()
{
    char* p;
    char a[20];
    int  i=0;
    p = f();
    while (*(p + i) != 0) {
        a[i] = p[i];
        i++;
    }
    a[i] = 0;
    printf("%s \n", a);
    printf("%s \n", p);
    return 0;
}
```



常见错误 返回局部地址

warning C4172: 返回局部变量或临时变量的地址: temp

```
    a[i] = 0;
    printf("%s \n", a);
    printf("%s \n", p);
    return 0;
```

监视 1

搜索(Ctrl+E) 搜索深度: 3

名称	值	类型
p	0x006ffb30 "hello"	char *

执行printf(%s\n",a) 前，
观察：p 指向的单元内容

```
    a[i] = 0;
    printf("%s \n", a);
    printf("%s \n", p);  已用时间 <= 1ms
    return 0;
```

监视 1

搜索(Ctrl+E) 搜索深度: 3

名称	值	类型
p	0x006ffb30 ""	char *

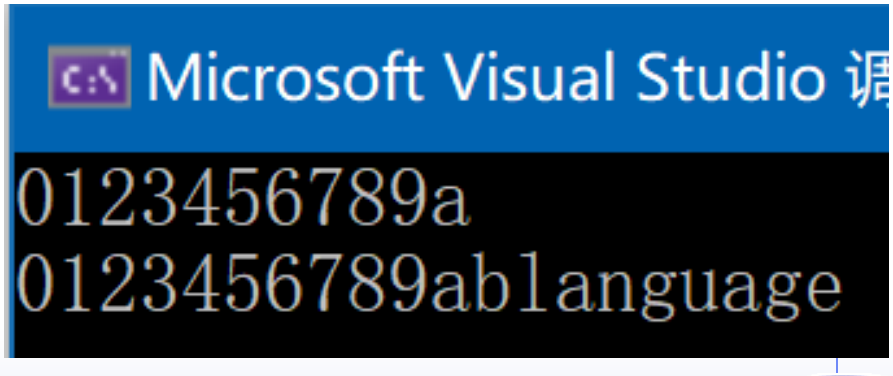
执行printf(%s\n",a) 后，
p 指向的单元未变
但单元串中内容发生变化

小测验——字符串访问


```
#include <iostream>
int main()
{
    char s1[20];
    char  s2[12];
    strcpy_s(s2, "0123456789a");
    printf("%s\n", s2);
    s2[11] = 'b';
    strcpy_s(s1, "language");
    printf("%s\n", s2);
    return 0;
}
```

字符串 以 0为结束符

Q：运行结果？

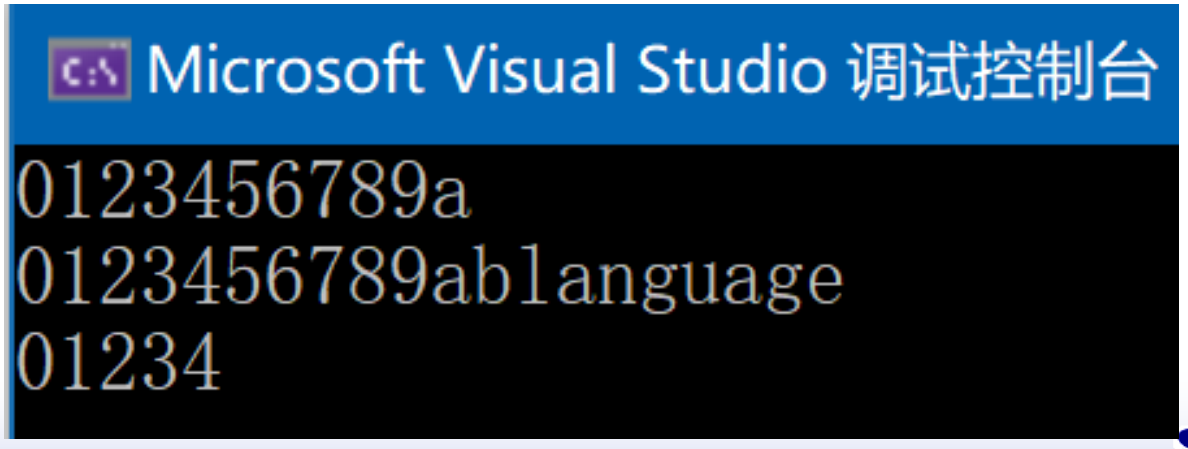


常见错误 字符串

```
#include <iostream>
int main()
{ ……
    printf("%s\n", s2);
    s2[5] = 0;
    printf("%s\n", s2);
    return 0;
}
```

字符串 以 0为结束符

Q：运行结果？

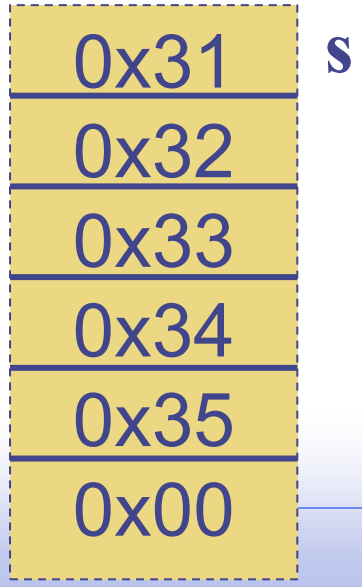


地址类型转换

```
char  s[10] = "12345";
 printf("%c  %x\n", *s, *s);
 printf("%d  %x\n", *(short *)s, *(short*)s);
 printf("%d  %x\n", *(int *)s, *(int*)s);
```

Q：运行结果是什么？ 为什么？

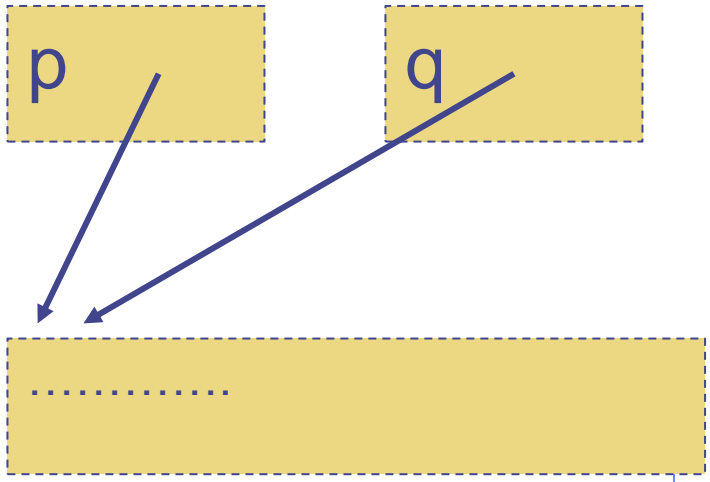
1　31
12849　3231
875770417　34333231



小测验——空间的分配与释

```
#include <iostream>
int main()
{
    char* p, *q;
    p =(char *)malloc(100);
    q = p;
    if (p != NULL) {
        free(p);        p = NULL;
    }
    if (q != NULL) {
        free(q);        q = NULL;
    }
    return 0;
}
```

Q：程序运行如何？



程序运行异常

空间的分配与释放


```
#include <iostream>
int main()
{
    char* p;
    p = (char*)malloc(100);
    strcpy_s(p, 6,"hello");
    return 0;
}
```

Q：程序存在何种问题？

内存泄露

空间的分配与释放

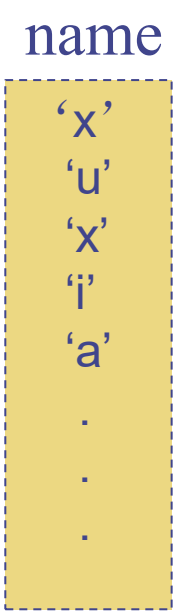

```
#define CRTDBG_MAP_ALLOC
#include <crtdbg.h>
#include <iostream>
int main()
{  char* p;
   p = (char*)malloc(100);
   strcpy_s(p, 6,"hello");
    _CrtDumpMemoryLeaks();
   return 0;
}
```

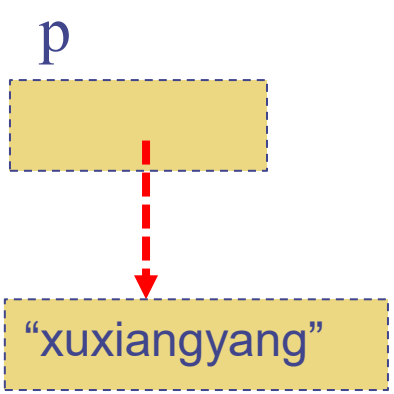
```
Detected memory leaks!
Dumping objects ->
{76} normal block at 0x01435130, 100 bytes long.
 Data: <hello           > 68 65 6C 6C 6F 00 CD CD CD CD CD CD CD CD CD CD
Object dump complete.
```

一维数组与指针


```
char  name[20]="xuxiangyag";
const char * p="xuxiangyang";
```

name[1] = ?
p[1] = ?
*(name+1) =?
*(p+1) =?





Q：能修改 &p 吗？ 不能

Q：能修改 name 吗？ 不能

一维数组与指针


```
char  name[20]="xuxiangyang";
const char * p="xuxiangyang";
```

Q：定义数组并赋初值，与定义指针并赋初值，有什么差别？

前者将字节流拷贝到数组空间,后者将串首地址送给 p

Q：设有如上定义，如下赋值语句语法是否正确？

```
name ="wang";        错误
 p="wang";           正确
```

Q：如下赋值语句语法是否正确？

```
name = p ;           错误
 p=name;             正确
```

常见错误

数组越界

返回函数中局部变量的地址

空间分配与释放不匹配

指针指向的地址错误