2019.M.01解析

信息学院　崔冠宇　20182021147

注：以下均为个人想法，若有不准确之处，敬请谅解。

解题步骤

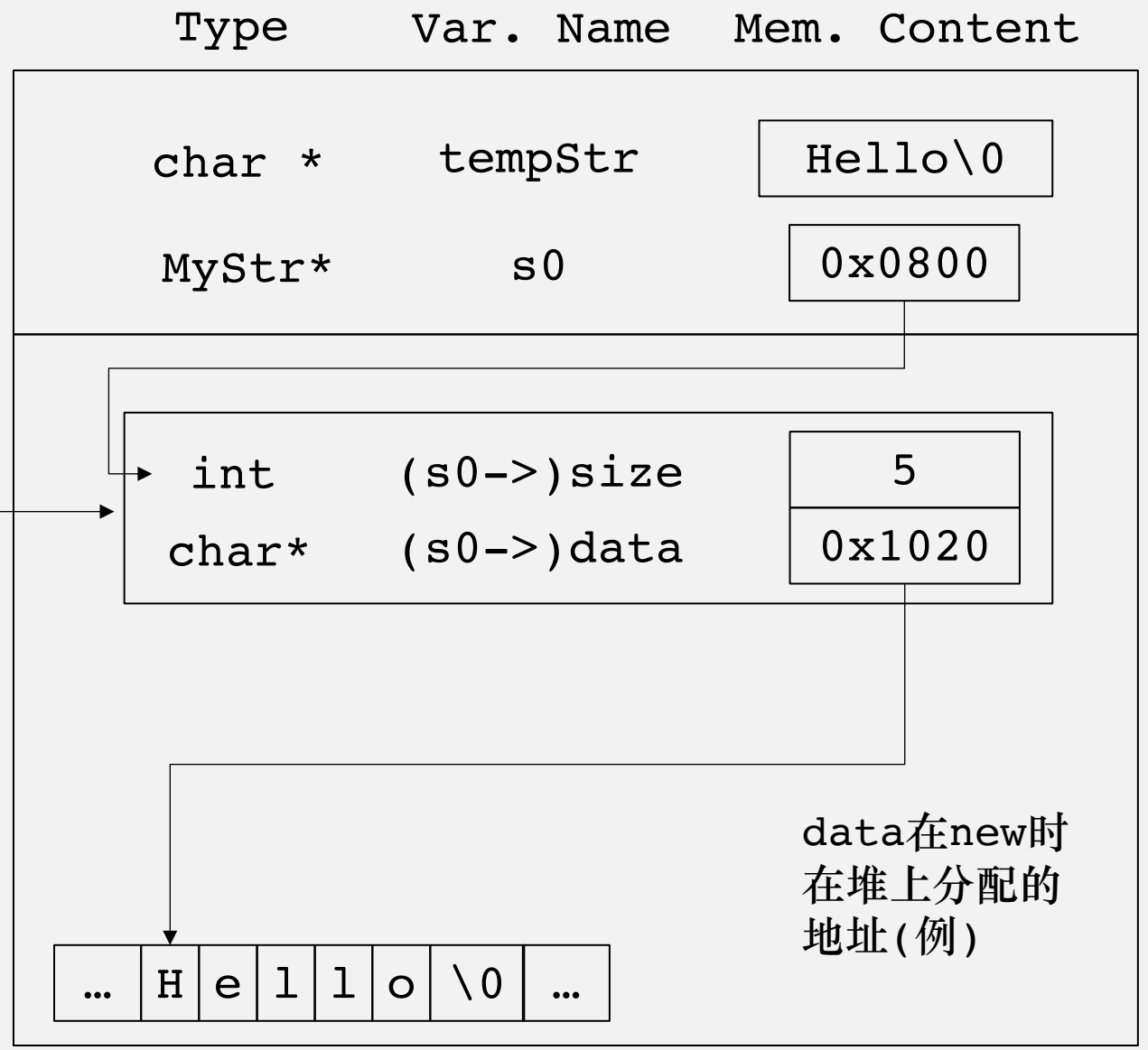
- 1.阅读题干，提取题目的要点：实现 MyStr 类.
- 2.阅读代码框架，先关注类的定义，并与main()比较，观察是否出现类定义中未写出的成员(可以看到这一行 “MyStr * s1 = new MyStr(*s0);” 需要自己写一个复制构造函数).
- 3.进一步阅读类的定义，搞清楚各个成员的含义与功能，并且找到需要自己补充的内容.（分析如下）
-

```
class MyStr{
public:
    MyStr(const char * s); //(非默认)构造函数，自行实现
    ~MyStr(); //析构函数，已给出实现
    void myStrcat(const MyStr & str1); //字符串连接，自行实现
    const char * getStr(); //返回字符串内容，已给出实现
//此处可补充代码. 根据分析，应当补充复制构造函数(定义)
private:
    int size; //长度
    char * data; //指向内容区段的指针
};
```

解题步骤

- 4. 在头脑中思考类和对象的数据组织形式，并且搞清楚main()中都做了什么.
- char tempStr[20];
  cin >> tempStr;
  //假定输入"Hello"
- MyStr * s0 = new MyStr(tempStr);

注：此处及以后地址均为示例，实际以运行结果为准。



解题步骤

- 5．含有指针的对象复制构造要小心！（深/浅拷贝）
- `MyStr * s1 = new MyStr(*s0);`

栈(Stack)空间

Type	Var. Name	Mem. Content
char*	tempStr	Hello\0
MyStr*	s0	0x0800
MyStr*	s1	0x0812

复制构造

int	(s0->)size	5
char*	(s0->)data	0x1020
int	(s1->)size	5
char*	(s1->)data	0x1080

0x1020

…	H	e	l	l	o	\0	…

0x1080

…	H	e	l	l	o	\0	…

注意：应当拷贝的是data指向的内容，地址应重新分配，不应浅拷贝

解题步骤

- 6.字符串连接和析构函数(猜想是为了保证评测服务器不发生内存泄漏，已给出具体实现)都要注意指针的问题.
- ```
  s1 -> myStrcat(*s0);
  cout << s1 -> getStr() << endl;
  s1 -> ~MyStr();
  cout << s0 -> getStr() << endl;
  s0 -> ~MyStr();
  ```

Type	Var. Name	Mem. Content
int	(s1->)size	10
char*	(s1->)data	0x2080

注意：字符串连接应重新分配空间

0x2080

…	H	e	l	l	o	H	e	l	l	o	\0	…
  ```

示例解答

```
//空位1
MyStr(const MyStr & s1);

//空位2
MyStr::MyStr(const char *s)
{
    if(s != NULL)
    {
        size = strlen(s);
        data = new char[size + 1];
        strcpy(data, s);
    }
    else size = 0, data = NULL;
}
```

（转右侧）

```
MyStr::MyStr(const MyStr &s1)
{
    if(s1.data != NULL)
    {
        size = s1.size;
        data = new char[size + 1];
        strcpy(data, s1.data);
    }
    else size = 0, data = NULL;
}
void MyStr::myStrcat(const MyStr &str1)
{
    char tmp[size + 1];
    strcpy(tmp, data);
    size += str1.size;
    delete[] data;
    data = new char[size + 1];
    strcpy(data, tmp);
    strcat(data, str1.data);
}
```