|  |
| --- |
| Tongji SSE |
| JSON库程序 |
| C语言课程项目 |
|  |
| **Lin, Ivan** |
| **12/1/2014** |

|  |
| --- |
| 同济大学软件学院C语言课程项目要求 |

目录

[1. 基本需求 2](#_Toc405219607)

[2. 具体需求 2](#_Toc405219608)

[2.1 数据结构 2](#_Toc405219609)

[2.2 接口 2](#_Toc405219610)

[2.3 说明 4](#_Toc405219615)

[3. 项目提交 5](#_Toc405219616)

[3.1 提交内容 5](#_Toc405219617)

[3.2 提交方式 6](#_Toc405219618)

[3.3 截止时间 6](#_Toc405219619)

[4. 评测 6](#_Toc405219620)

[4.1 基本要求 6](#_Toc405219621)

[4.2 自动化评测 6](#_Toc405219622)

[4.3 代码质量 7](#_Toc405219623)

[4.4 其他 7](#_Toc405219624)

[5. 附录 7](#_Toc405219625)

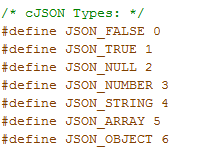
## 基本需求

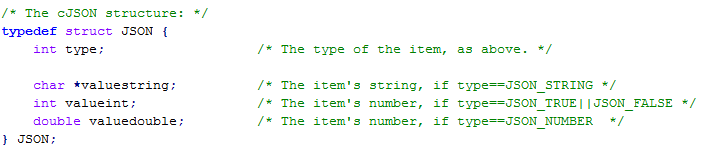
个人项目：编写一个库程序能够提供指定接口对JSON数据进行操作。

## 具体需求

本项目将提供一个.h头文件，里面定义了JSON的数据结构以及需要实现的函数。各位同学需要实现这些接口（**可自行增加需要的函数，但是不能删除给定的头文件里的函数**），之后评测程序将来验证这些接口的是否符合需求。

### 数据结构





大家需要**严格按照上面的声明来定义基本数据结构（上面的这些成员及type定义是必须的）**，以方便最终的测试。上述JSON定义只能覆盖到除Array和Object外的基本类型，其他JSON内成员可以根据自己实现的需要**自由添加**。

### 接口

|  |  |  |
| --- | --- | --- |
| 函数声明 | 函数说明 | 参数说明 |
| Parse & Print | | |
| JSON \*ParseJSON(const char \*value); | 解析JSON字符串 | value为需要解析的字符串，函数返回指向解析后的JSON的指针。 |
| JSON \*ParseJSONFromFile(const char \*file\_name); | 解析JSON文件 | file\_name为需要解析的文件名，函数返回指向解析后的JSON的指针。 |
| void PrintJSON(JSON \*item); | 将JSON**无格式**输出到屏幕。 | item指向需要输出的JSON。 |
| void PrintJSONToFile(JSON \*item, const char \*file\_name); | 将JSON按**指定格式**输出到文件中。 | Item指向需要输出的JSON，file\_name为文件名。 |
| Create | | |
| JSON \*CreateNULL(void); | 创建一个NULL。 | 返回一个指向创建后的JSON的指针。 |
| JSON \*CreateTrue(void); | 创建一个True。 | 返回一个指向创建后的JSON的指针。 |
| JSON \*CreateFalse(void); | 创建一个False。 | 返回一个指向创建后的JSON的指针。 |
| JSON \*CreateBool(int b); | 创建一个True或False。 | b为0则创建False，否则创建True。返回一个指向创建后的JSON的指针。 |
| JSON \*CreateNumber(double num); | 创建一个Number。 | num为值，返回一个指向创建后的JSON的指针。 |
| JSON \*CreateString(const char \*string); | 创建一个String。 | string为值，返回一个指向创建后的JSON的指针。 |
| JSON \*CreateArray(void); | 创建一个Array。 | 返回一个指向创建后的JSON的指针。 |
| JSON \*CreateObject(void); | 创建一个Object。 | 返回一个指向创建后的JSON的指针。 |
| Append | | |
| void AddItemToArray(JSON \* array, JSON \*item); | 将一个JSON添加到Array中。 | 将item指向的JSON添加到array指向的JSON中。如原来array为[1, 2, 3]，添加进一个type为JSON\_TRUE的item，则array变为[1, 2, 3, true]。 |
| void AddItemToObject(JSON \*object, const char \*key, JSON \*value); | 讲一个JSON添加到Object中。 | 将item指向的JSON添加到object指向的JSON中，并且key为指定值。如原来object为{ “name”: “c”}，添加进一个type为JSON\_TRUE的item，key为”is\_girl”，则object变为{“name”: “c”, “is\_girl”: “true”} |
| Update | | |
| void ReplaceItemInArray(JSON \*array, int which, JSON \*new\_item); | 替换array中的一个item。 | which为需要替换的item是array中第几个，从0开始。 |
| void ReplaceItemInObject(JSON \*object, const char \*key, JSON \*new\_value); | 替换object中的一个item。 | key为需要替换的键，值替换为new\_value指向的JSON。 |
| Remove/Delete | | |
| JSON \*DetachItemFromArray(JSON \*array, int which); | 将array中的某个元素删除，不释放其内存。 | array指向一个JSON Array，which表示要删除的元素的下标，从0开始。返回指向被删除元素的指针。 |
| void \*DeleteItemFromArray(JSON \*array, int which); | 将array中的某个元素删除，释放其内存。 | array指向一个JSON Array，which表示要删除的元素的下标，从0开始。 |
| JSON \*DetachItemFromObject(JSON \*object, const char \*key); | 将object中的某个元素删除，不释放其内存。 | object指向一个JSON Object，key表示要删除的元素的key。返回指向被删除元素的指针。 |
| void \*DeleteItemFromObject(JSON \*object, const char \*key); | 将object中的某个元素删除，释放其内存。 | object指向一个JSON Object，key表示要删除的元素的key。 |
| void DeleteJSON(JSON \*item); | 删除一个JSON，释放内存。 | Item指向需要删除的JSON。 |
| Duplicate | | |
| JSON \*Duplicate(JSON \*item, int recurse); | 复制一个JSON。 | Item指向需要被复制的JSON，recurse表示是否**深度复制**。 |
| Read | | |
| JSON \*GetItemInArray(JSON \*array, int which); | 获取array中某个元素。 | Which表示需要获取的元素的下标，从0开始。 |
| JSON \*GetItemInObject(JSON \*object, const char \*key); | 获取object中某个元素。 | Key表示需要获取的元素的键。 |
| JSON \*GetItemInJSON(JSON \*json, const char \*path); | 按路径获取json中的某个元素。 | Path表示路径，如下面这个JSON  {  “school”: “Tongji”,  “properties”: {  “age”: 18,  “courses”: [  “C language”,  “C++ and Java”  ]  }  }  那么，/school的值为”Tongji”,  /properties/courses/1的值为”C++ and Java” |



### 说明

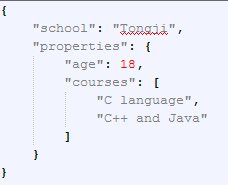
* **输出**：所有key需要加上双引号，value格式参考[JSON标准](#_附录)。JSON\_TRUE和JSON\_FALSE类型分别输出true和false（小写）,JSON\_NULL输出null（小写）。JSON\_STRING要求有双引号，JSON\_NUMBER不需要双引号。
* **无格式**：除string中的空格之外不能存在其他空格。

如：

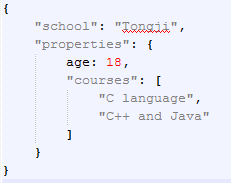
[true,false,"111",{"fff":"fff"}]

{"age":20}

* **指定格式**：如果JSON为object或者array类型，那么首尾的 '{' 或 '[' 需要各占一行，其余的 '{' 或 '[' 需要在 ':' 后一个空格输出。所有缩进统一为**四个空格**。例子如下：



* **JSON文件举例**（给定的JSON文件不一定符合指定格式，如样例文件中age就没有引号，但这样的读入是合法的）：



* **路径**：从最外层JSON开始，以’/’分隔，类似于文件路径，数组中元素用下标定位。如上述JSON，/school的值为”Tongji”，/properties/age的值为18，/properties/courses/1的值为”C++ and Java”。
* **深度复制**：递归复制，即创建一个全新的JSON，不引用原来JSON中的任何部分。

## 项目提交

### 提交内容

* JSON.h 包含了自己修改后的struct JSON 和 要求提供的函数
* JSON.c 实现JSON.h中的函数及其他自定义内容
* 其他头文件及c文件
* 项目说明文档（大概介绍一下开发过程以及收获，如果有实现其他头文件及c文件请说明用法）

### 提交方式

### 截止时间

## 评测

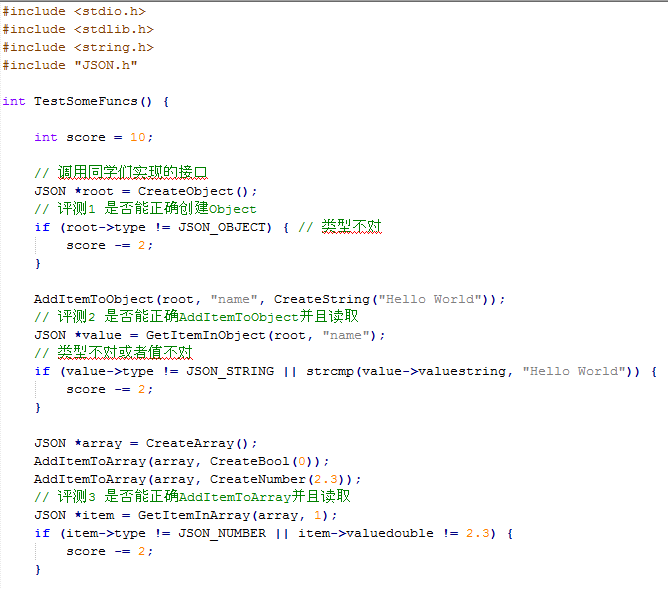
### 基本要求

实现头文件中指定接口，并且程序能够运行。

### 自动化评测

我们将通过程序调用你实现好的函数，通过与预期结果对比实现自动化评测部分。

举例如下（实际测试时测试函数不会这么大，粒度会更小）：





### 代码质量

* 代码风格，变量命名、函数命名、缩进等等。
* 是否有大量冗余、重复
* 注释

### 其他

* 内存使用及管理情况
* \*健壮性

## 附录

* [JSON标准](http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf)
* [JSON在线格式化](http://tool.oschina.net/codeformat/json)
* JSON.h
* example.json
* test.c