1 详细阐述HTTP协议中POST请求和PUT请求的语义差别。

参考答案

根据HTTP协议规定，POST和PUT请求都可以用于向服务器传递数据，并请求服务器将数据保存起来，故所有的请求消息中只有这二者可以有请求消息主体。

但二者有细微的差别：PUT方法和GET、HEAD、DELETE一样都属于等幂方法（Idempotent Mehtods）—— 等幂就是值不变性，相同的请求得到相同的响应结果，不会出现相同的请求出现不同的响应结果。例如针对某一URI提交相应的修改数据，不论执行多少次都应该是得到相同的数据结果。而POST方法属于非等幂方法，针对同一URI提供相同的数据，每次都应产生不同的结果，如添加操作，可能每次POST提交都可能产生一条新的数据。

2 举例说明Express中客户端向服务器提交数据的方法有哪些，Express中如何读取客户端提交的数据。

参考答案

```
01.    //客户端请求消息起始行：
02.    GET /user?uid=15 HTTP/1.1
03.
04.    //服务器端Express应用可以使用get/delete路由方法接收请求数据：
05.    app.get('/user', function(req, res){
06.                    console.log( req.query.uid );                    //15
07.    });
08.
09.
10.    //客户端请求消息起始行：
11.    POST /user HTTP/1.1
12.    ...
13.    uname=tom&upwd=123456
14.
15.    //服务器端Express应用可以使用post或put路由方法通过解析请求主体读取请求数
16.    app.post('/user', function(req, res){
17.                    req.on('data', function(data){
18.                            var obj = querystring.parse(data.toString());
19.                            console.log(obj.uname);                    //tom
20.                            console.log(obj.upwd);                    //123456
21.    });
22.    });
```

```
23.
24.
25.
26.     //客户端请求消息起始行：
27.     GET /list/3/regTime HTTP/1.1
28.
29.     //服务器端Express应用可以在对应路由方法中通过请求对象的params属性读取路由
30.     app.get('/list/:pno/:orderBy', function(req, res){
31.                     console.log(obj.params.pno);                          //3
32.                     console.log(obj.params.orderBy);                  //regTime
33.     });
34.
35.     //客户端请求消息起始行：
36.     GET /user HTTP/1.1
37.     Cookie: userid=99E7EE368AC0E0FA9F159C7D420D29C9; SESSID= 1420_21120_17001
38.
39.     //服务器端Express应用可以在对应路由方法中通过解析请求对象的headers属性中的
40.     app.get('/list/:pno/:orderBy', function(req, res){
41.                     req.headers.cookie;
42.                     req.cookies;
43.     });
```

3 自定义一个adminLoginCheck中间件，若客户端请求了/admin/*目录下的资源但未提供有效的LOGIN_TOKEN，则会被强制跳转到/login页面。

假设LOGIN_TOKEN（登录之后获取的令牌）就是一个随意指定的很长的随机字符串，只要登录完成后，服务器就会把此令牌告知给客户端，客户端只要在接下来的访问中出示此令牌，即视为登录后的有效管理员。

参考答案

创建一个声明了中间件函数的模块文件，middleware.js，代码如下：

```
01.     function adminLoginCheck(){
02.       return function(req, res, next){
03.         if( req.query.login_token ){     //此处省略了对LOGIN_TOKEN的更复杂的验
04.           next();
05.         }else {
06.           res.redirect('/login');
07.         }
```

```
    }
  }

  module.exports.adminLoginCheck = adminLoginCheck;
```

创建应用程序的主启动文件index.js，使用刚刚定义中间件模块，代码如下：

```
const express = require('express');
const middleware = require('./middleware');

var app = express();
app.listen(80);

app.use('/admin/*', middleware.adminLoginCheck());
app.get('/admin/list', function(req, res){
  res.send('admin list ...');
})
app.get('/login', function(req, res){
  res.send('admin login ...');
})
```