Разработка программы для автоматического извлечения данных с сайта

Буев А.А.

Аннотация.
Файл является описанием проекта, выполненного в рамках научно-исследовательского семинара "Введение в специальность".

Автор:
Буев Антон Александрович,
НИУ ВШЭ МИЭМ им. А.Н.Тихонова
ул. Таллинская улица, 34, г.Москва, Россия, 123458
e-mail: aabuev@edu.hse.ru

Введение

Целью данной работы является разработка программы, автоматически извлекающей данные о патентах с web-сайта **(US Patent & Trademark Office, Patent Full Text and Image Database)**.
Метод решения поставленной задачи описан с помощью языка программирования `C` (в среде разработки *Visual Studio 2015*).
В первой главе рассматривается общий алгоритм работы программы, подробное описание кода находится во второй главе,в третьей обсуждаются полученные в ходе результаты; в заключении содержатся выводы.

1. Описание решения

На сайте **(US Patent & Trademark Office, Patent Full Text and Image Database)** имеется файл *Patent Grant Authority File* (ссылка на скачивание), содержащий все номера патентов, зарегестрированных в США.
Чтобы автоматически скачивать данные с сайта, была разработана программа, которая:

1. Берет номер из *Patent Grant Authority File*
2. Создает сокет, необходимый для обмена данными с сервером, подключается к серверу
3. Отправляет HTTP GET запрос, используя взятый номер патента
4. Обрабатывает полученный от сервера ответ
5. Закрывает сокет и возвращается к пункту 1.

2. Детальный разбор

Часть 1. int main()

Устраняем возможные ошибки, покдлючаем библиотеки, пишем прототипы функций.

Листинг 1. библиотеки и прототипы функций

```
#define _CRT_SECURE_NO_WARNINGS
#define _WINSOCK_DEPRECATED_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <malloc.h>
#include <winsock2.h>
#include <ws2tcpip.h>
#include <windows.h>
#include <time.h>
#pragma comment(lib,"ws2_32.lib")

char* GetPNumber(char*);
char* MessageCreate(char *, char *);
char* modify(char*, long);
void parse(FILE*, char*, long);
char* modifyClassNumber(char*, long);
void MakeDelay();
long LoadLastNum();
void SaveLastNum(long);
void error(const char *msg) {perror(msg); exit(0);}
```

Объявляем и инициализируем необходимые переменные. Указываем адрес хоста. Создаем папку, в которую будет загружаться информация о патентах. Открываем файл *Patent Grant Authority File*, сокращенно *PGA.txt*, и смещаемся в нем на значение `offset`, которое было загружено с помощью функции `LoadLastNum()`(ссылка на описание 2.) из файла *log.bin*, в который функция `SaveLastNum(offset)`(ссылка на описание 2.) сохранила последнее смещение.

Листинг 2. `int main()` 1-ая часть

```
23 int main()
24 {
25     int received;
26     char FolderName[40], FileName[40], line[50], *PNumber, buff[1025];
27     long offset = LoadLastNum();
28     char message[256] = { 0 };
29     char host[] = "patft.uspto.gov";
30
31     _mkdir("E:\\Patents");
32     FILE* PGA = fopen("PGA.txt", "r");
33     if (PGA == NULL)
34     {
35         printf("Couldn't open PGA file...\n");
36         return -1;
37     }
38     fseek(PGA, offset, SEEK_SET);
```

Объявляем переменные необходимые для работы с сокетом, инициализируем его.

Листинг 3. `int main()` 2-ая часть

```
40     int portno = 80;
41     struct hostent *server;
42     struct sockaddr_in serv_addr;
43     WSADATA wsa;
44     SOCKET s;
45     printf("\nInitialising Winsock...");
46     if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
47     {
48         printf("Failed. Error Code : %d", WSAGetLastError());
49         return 1;
50     }
51     printf("Initialised.\n");
52     server = gethostbyname(host);
53     serv_addr.sin_addr.s_addr = inet_addr(server->h_addr);
54     memset(&serv_addr, 0, sizeof(serv_addr));
55     serv_addr.sin_family = AF_INET;
56     serv_addr.sin_port = htons(80);
57     memcpy(&serv_addr.sin_addr.s_addr,server->h_addr,server->h_length);
```

Делаем цикл, который будет выполняться, пока номера патентов из файла *PGA.txt* не закончатся. В нем мы создаем сокет, управляем соединением, достаем номер патента `PNumber = GetPNumber(line)`(ссылка на описание 2.), создаем HTTP GET запрос в отдельной функции `MessageCreate(PNumber, message)`(ссылка на описание 2.) и отправляем его.

Листинг 4. `int main()` 3-ая часть; цикл 1-ая часть

```
while (!feof(PGA))
{
    if ((s = socket(AF_INET, SOCK_STREAM, 0)) == INVALID_SOCKET)
    {
        printf("Could not create socket : %d", WSAGetLastError());
    }
    if (connect(s,(struct sockaddr *)&serv_addr, sizeof(serv_addr))<0)
    {
        printf("connect failed with error code: %d",WSAGetLastError());
            return 1;
    }
    fgets(line, 50, PGA);
    offset += strlen(line);
    PNumber = GetPNumber(line);
    printf("\nPNumber: %s\n", PNumber);
    MessageCreate(PNumber, message);
    if (send(s, message, strlen(message), 0) < 0)
    {
        printf("Send failed with error code : %d", WSAGetLastError());
        return 1;
    }
```

Создаем папку для данного номера патента, куда помещаем файл `Origin.txt`, в котором будет храниться ответ, который мы получаем с сервера.

Листинг 5. `int main()` 4-ая часть; цикл 2-ая часть

```
    sprintf(FolderName, "E:\\Patents\\%s", PNumber);
    _mkdir(FolderName);
    sprintf(FileName, "E:\\Patents\\%s\\Origin.txt", PNumber);
    FILE * Origin = fopen(FileName, "w");
    if (Origin != NULL)
    {
        do {
            received = recv(s, buff, 1024, 0);
            if (received > 0)
            {
                buff[received < 1024 ? received : 1024] = '\0';
                fprintf(Origin, "%s", buff);
            }
            else if (received == 0)
                printf("Connection closed");
            else
                printf("recv failed: %d\n", WSAGetLastError());
        } while (received > 0);
```

Определяем размер файла `Origin.txt`. Вызываем функцию `parse(Origin, PNumber, size)`(ссылка на описание 2.). Она обработает ответ, полученный от сервера, так как в ответе присутствует HTML разметка, а также разделит файл на 4 файла(`Abstract.txt, Class number.txt, Claims.txt, Description.txt`) и сохранит их в той же папке. Затем закроется сокет, удалится файл `Origin.txt`, сохранится смещение, для того, чтобы при повторном открытии программы началось скачивание с последнего номера. Также реализована функция `MakeDelay()`(ссылка

на описание 2.), которая делает случайную паузу в 1-3 секунды для того, чтобы сервер не заблокировал за превышение количетсва запросов. Конец цикла.

Листинг 6. `int main()` 5-ая часть; цикл 3-ая часть

```
98          freopen(FileName, "r", Origin);
99          fseek(Origin, 0, SEEK_END);
100         long size = ftell(Origin);
101         fseek(Origin, 0, SEEK_SET);
102         if (size > 0)
103         {
104             parse(Origin, PNumber, size);
105         }
106         else printf("FILE is empty\n");
107         fclose(Origin);
108     }
109     else printf("ERROR creating a file\n");
110     closesocket(s);
111     remove(FileName);
112     SaveLastNum(offset);
113     MakeDelay();
114 }
```

Конец функции `int main()`.

Листинг 7. `int main()` 6-ая часть; цикл 3-ая часть

```
115     fclose(PGA);
116     WSACleanup();
117     return 0;
118 }
```

Часть 2. Остальные функции

Функция, вызывающая паузу.

Листинг 8. `void MakeDelay()`

```
120 void MakeDelay()
121 {
122     int randNum;
123     srand(time(NULL));
124     randNum = rand() % 3 + 1;
125     printf("\nSleep for %d seconds\n", randNum);
126     Sleep(randNum * 1000);
127 }
```

Функции, запоминающие номер.

Листинг 9. Загружает смещение `long LoadLastNum()`

```
129 long LoadLastNum()
130 {
131     long offset = 0;
132     FILE* log = fopen("log.bin", "rb");
```

```
133     if (log != NULL)
134     {
135         fread(&offset, sizeof(long), 1, log);
136         fclose(log);
137     }
138     return offset;
139 }
```

Листинг 10. Сохраняет смещение `void SaveLastNum(long offset)`

```
141 void SaveLastNum(long offset)
142 {
143     FILE* log = fopen("log.bin", "wb");
144     if (log != NULL)
145     {
146         fwrite(&offset, sizeof(long), 1, log);
147         fclose(log);
148     }
149     else printf("Couldn't open log.bin file\n");
150 }
```

Функция, возвращающая строку в форме HTML GET запроса, который отправляется на сервер.

Листинг 11. `char* MessageCreate(char * PN, char * message)`

```
152 char* MessageCreate(char * PN, char * message)
153 {
154     char host[] = "patft.uspto.gov";
155     char path[256] = "netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL
    &p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=";
156     strcat(path, PN);
157     strcat(path, ".PN.&OS=PN/");
158     strcat(path, PN);
159     strcat(path, "&RS=PN/");
160     strcat(path, PN);
161     sprintf(message, "GET /%s HTTP/1.1\r\nHost: %s\r\nContent-Type:
    text/html\r\n\r\n", path, host);
162     return message;
163 }
```

Функция, обрабатывающая строку, взятую из *PGA.txt*, возвращает номер патента в виде строки.

Листинг 12. `char* GetPNumber(char * line)`

```
165 char* MessageCreate(char * PN, char * message)
166 {
167     char host[] = "patft.uspto.gov";
168     char path[256] = "netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL
    &p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=";
169     strcat(path, PN);
170     strcat(path, ".PN.&OS=PN/");
171     strcat(path, PN);
172     strcat(path, "&RS=PN/");
```

```
173     strcat(path, PN);
174     sprintf(message, "GET /%s HTTP/1.1\r\nHost: %s\r\nContent-Type:
        text/html\r\n\r\n", path, host);
175     return message;
176 }
```

Функция, обрабатывающая файл `Origin.txt`, создавая 4 файла(`Abstract.txt`, `Class number.txt`, `Claims.txt`, `Description.txt`) и помещая в них обработанную информацию, соответствующую названиям. Поиск осуществляется, путем помещения файла `Origin.txt` в строку, а затем с помощью стандартной функции `strstr` по тегам ищется необходимая часть, измеряется размер этой части в байтах и копируется в новую строку.

Листинг 13. `void* parse(FILE * origin, char * PN, long size)`

```
188 void parse(FILE* origin, char* PN, long size)
189 {
190     char abstractName[40];
191     char numbersName[40];
192     char claimsName[40];
193     char descriptionName[40];
194     sprintf(abstractName, "E:\\Patents\\%s\\Abstract.txt", PN);
195     sprintf(numbersName, "E:\\Patents\\%s\\Class number.txt", PN);
196     sprintf(claimsName, "E:\\Patents\\%s\\Claims.txt", PN);
197     sprintf(descriptionName, "E:\\Patents\\%s\\Description.txt", PN);
198     FILE * abstract = fopen(abstractName, "w");
199     FILE * numbers = fopen(numbersName, "w");
200     FILE * claims = fopen(claimsName, "w");
201     FILE * description = fopen(descriptionName, "w");
202 if ((abstract!=NULL)&&(numbers!=NULL)&&(claims!=NULL)&&(description!=
    NULL))
203     {
204         char* html = (char*)malloc(size + 1);
205         if (html != NULL)
206         {
207             size_t nread = fread(html, 1, size, origin);
208     char* abstr=strstr(html,"<BR><CENTER><b>Abstract</b></CENTER>");
209             if (abstr != NULL)
210             {
211                 char* beg = strstr(abstr, "<p>");
212                 char* end = strstr(beg, "</p>");
213                 long beglen = strlen(beg);
214                 long endlen = strlen(end);
215                 long bodylen = beglen - endlen;
216                 char *strBODY = (char*)malloc(bodylen + 1);
217                 memcpy(strBODY, beg, bodylen);
218                 fprintf(abstract, "%s", modify(strBODY, bodylen));
219                 free(strBODY);
220             }
221     char* clss=strstr(html,"<b>Current International Class: </b>");
222             if (clss != NULL)
223             {
224                 char* beg = strstr(clss, "</TD>");
225                 char* end = strstr(beg, "</TR>");
226                 long beglen = strlen(beg);
```

```
227                 long endlen = strlen(end);
228                 long bodylen = beglen - endlen;
229                 char *strBODY = (char*)malloc(bodylen + 1);
230                 memcpy(strBODY, beg, bodylen);
231         fprintf(numbers, "%s", modifyClassNumber(strBODY, bodylen));
232                 free(strBODY);
233             }
234     char* clms=strstr(html,"<CENTER><b><i>Claims</b></i></CENTER>");
235             if (clms != NULL)
236             {
237                 char* beg = strstr(clms, "<BR><BR>");
238                 char* end = strstr(beg, "<HR>");
239                 long beglen = strlen(beg);
240                 long endlen = strlen(end);
241                 long bodylen = beglen - endlen;
242                 char *strBODY = (char*)malloc(bodylen + 1);
243                 memcpy(strBODY, beg, bodylen);
244                 fprintf(claims, "%s", modify(strBODY, bodylen));
245                 free(strBODY);
246             }
247 char* dscrp=strstr(html,"<CENTER><b><i>Description</b></i></CENTER>");
248             if (dscrp != NULL)
249             {
250                 char* beg = strstr(dscrp, "<BR><BR>");
251                 char* end = strstr(beg, "<HR>");
252                 long beglen = strlen(beg);
253                 long endlen = strlen(end);
254                 long bodylen = beglen - endlen;
255                 char *strBODY = (char*)malloc(bodylen + 1);
256                 memcpy(strBODY, beg, bodylen);
257                 fprintf(description, "%s", modify(strBODY, bodylen));
258                 free(strBODY);
259             }
260             free(html);
261         }
262         else printf("can't allocate mem for html file\n");
263 fclose(abstract), fclose(numbers), fclose(claims), fclose(description);
264     }
265     else printf("ERROR 1\n");
266 }
```

Функция, обрабатывающая строку, содержащую *Class Number*, но с HTML разметкой и набором символов , обозначающих неразрывный пробел, возвращает номер класса патента в виде строки.

Листинг 14. `char* modifyClassNumber(char * string, long size)`

```
268 char* modifyClassNumber(char* string, long size)
269 {
270     char* tmp = (char*)malloc(size + 1);
271     long i = 0, j = 0, YoN = 0;
272     for (; i < size; i++)
273     {
274         if (string[i] == '<')
275         {
```

```
276             do {
277                 i++;
278             } while (string[i] != '>');
279         }
280         else if (string[i] == '&')
281         {
282             i += 4;
283         }
284         else if (string[i] == '\0')
285             break;
286         else tmp[j++] = string[i];
287     }
288     tmp[j] = '\0';
289     return tmp;
290 }
```

Функция, возвращающая строку, без HTML разметки.

Листинг 15. `char* modify(char * string, long size)`

```
292 char* modify(char* string, long size)
293 {
294     char* tmp = (char*)malloc(size + 1);
295     long i = 0, j = 0, YoN = 0;
296     for (; i < size; i++)
297     {
298         if (string[i] == '<')
299         {
300             do {
301                 i++;
302             } while (string[i] != '>');
303         }
304         else if (string[i] == '\0')
305             break;
306         else tmp[j++] = string[i];
307     }
308     tmp[j] = '\0';
309     return tmp;
310 }
```

3. Результаты

Пример скачанной информации.
4 файла находятся в одной папке с номером патента. Файлы сформированы в соответствии с разделами информации на сайте.
Номер в данном примере - 3922799(**ссылка на сайт**).

Раздел **Abstract**. Содержит краткое описание патента.

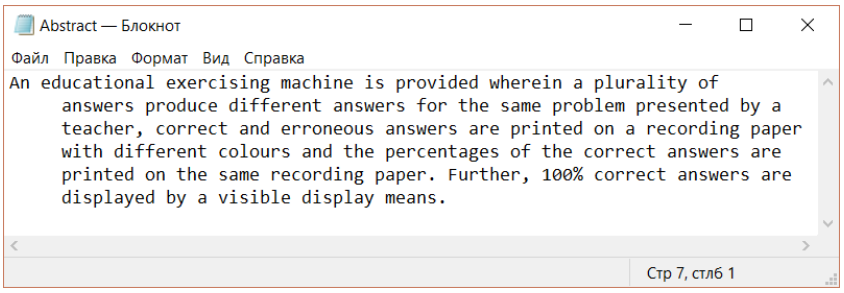

Рис. 1. `Abstract.txt`

Далее на сайте идет раздел с различными номерами классов. Данная программа скачивает только **Current International Class**.

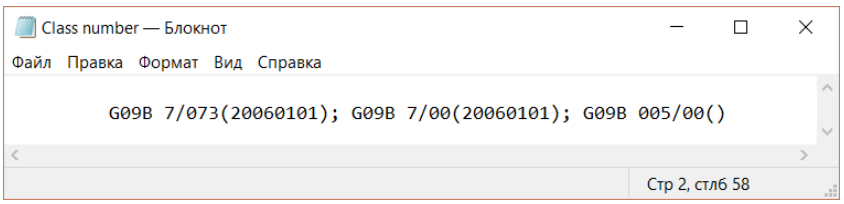

Рис. 2. `Class number.txt`

Раздел **Claims**. Описывается область и способы применения патента.

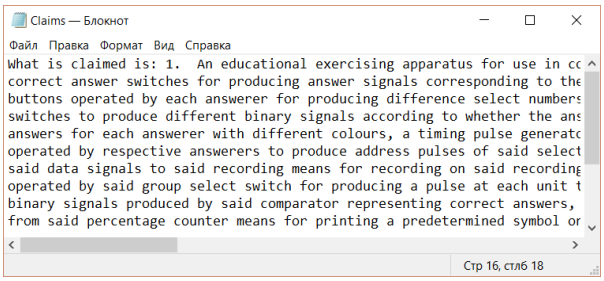

Рис. 3. `Claims.txt`

Раздел **Description**. Содержит более детальное описание патента, предысторию.

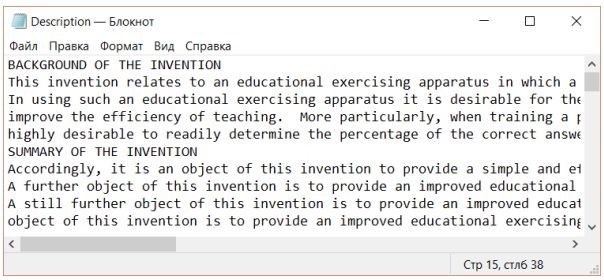

Рис. 4. `Description.txt`

Папка со всеми скачанными за 7-8 часов работы программы патентами:

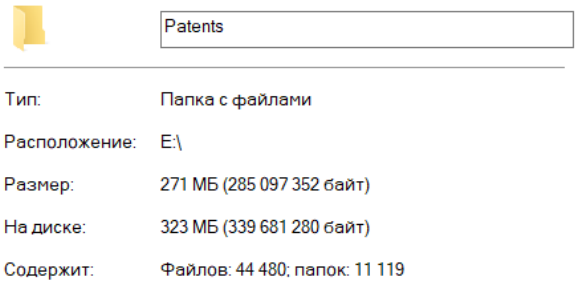

Рис. 5. `E:\Patents`

4. Заключение

Программа разработанна и успешно работает. Автоматизация процесса налажена. При необходимость программа может быть приспособленна к получению информации с других web-сайтов.

Список литературы / References

[1] Щур Л. Н., “Система LaTex для подготовки научных публикаций: учебно-методическое пособие”, *М.: МИЭМ НИУ ВШЭ*, 2017, 27.

[2] Брайан Кернига, Деннис Ритчи, “Язык программирования С, 2-е издание”, 2012.

[3] Генерация HTTP запросов, “URL: http://www.codenet.ru/webmast/php/http-post.php”.

[4] Winsock для всех, “URL: https://club.shelek.ru/viewart.php?id=35”.