a. Value Types Associated with Terminals and Non-Terminals

```
union{
        int ival;
        char *str;
}
```

b. Terminals

Regular Expression	Token	Type
[<](div)[ ](class)[=].(caption)[_](text).[>]	DIVON	String
[<](div)[^>]*[>]	DIVO	String
[<][/](div)[>]	DIVC	String
[<][h](0l1l2l3l4l5l6)([^>])*[>]	HO	String
[<][/][h](0l1l2l3l4l5l6)[>]	HC	String
[<][p]([^>])*[>]	PO	String
[<][/][p][>]	PC	String
[<](span)([^>])*[>]	SO	String
[<][/](span)[>]	SC	String
[<](ul)([^>])*[>]	ULO	String
[<][/](ul)[>]	ULC	String
[<](li)([^>])*[>]	LIO	String
[<][/](li)[>]	LIC	String
[<][a]([^>])*[>]	AO	String
[<][/][a][>]	AC	String
[<]i[ ](class)[=].(fa[ ]fa[-]envelope).[>][<][/]i[>]	IOE	String
[<]i[ ](class)[=].(fa[ ]fa[-]phone).[>][<][/]i[>]	IOP	String
[<]i[ ](class)[=].(fa[ ]fa[-]file[-]text).[>][<][/]i[>]	IOW	String
[<](div)[ ](class)[=].(researcharea).[>]	DIVOR	String
[<](div)[ ](class)[=].(accordion)[-](contact)[-](list).[>]	DIVOC	String
[<](div)[ ](class)[=].(facultyleftTabScroll)[^>]*[>]	DIVOFTS	String
[+a-zA-Z0-9][-+a-zA-Z@0-9&()/;,.]*[a-zA-Z0-9).]	string	String
[<](!--)	COMMENTO	Integer
(--)[>]	COMMENTC	Integer
[<](link)([^>])*[>]	LINK	Integer

c. Non-Terminals

Non Terminal	Type
LS	Integer

d. Start Symbol

T

e. Rules Of the Grammar

Rules	Action
DN : DIVON HO string HC PO T SO string SC SO string SC PC DIVC	fprintf(designation, "%s;%s\n", $3, $8); strcpy(name,$3);
LS : LS LIO string LIC	strcpy(&list[$1], $3); $$ = $1 + strlen($3); list[$$++] = 'l'; list[$$] = '\0';
LS : LS LIO LIC	$$ = $1;
LS :	$$ = 0;
EMAIL : LIO IOE string LIC	char *ptr = strstr($3, ";"); fprintf(email, "%s;%s\n",name, ptr + 1);
EMAIL :	Null
PHONE : LIO IOP string LIC	char *ptr = strstr($3, ";"); fprintf(phone, "%s;%s\n",name, ptr + 1);
PHONE :	Null

R1 : DIVO ULO LS ULC DIVC	int i = 0;
	fprintf(responsibility, "%s;", name);
	while(i < $3)
	  fprintf(responsibility, "%c", list[i++]);
	fprintf(responsibility, "\n");

R1 : DIVOC ULO EMAIL PHONE ULC DIVC	Null
RESCON : DIVOR DIVO DIVO DIVO DIVO T DIVC DIVO R1 DIVC DIVC DIVC C DIVC DIVC	Null
RESCON :	Null

Rules	Action

WEB : HO IOW AO string AC AO string AC HC	char *ptr = strstr($6, "href") + 6;
	int i = 0;
	char temp[100];
	while(*ptr != '\"'){
	temp[i++] = *ptr;
	ptr++;
	}
	temp[i] = '\0';
	fprintf(website, "%s;%s\n",name, temp);

WEB : HO IOW AO string AC HC	char *ptr;
	int i = 0;
	char temp[100];
	if(strstr($4, "Bio Sketch") == 0){
	ptr = strstr($3, "href") + 6;
	while(*ptr != '\"'){
	temp[i++] = *ptr;
	ptr++;
	}
	temp[i] = '\0';
	fprintf(website, "%s;%s\n",name, temp);
	}

WEB :	Null

R2 : DIVO DIVO ULO LS ULC DIVC DIVC	int i = 0;
	fprintf(awards, "%s;", name);
	while(i < $4)
	fprintf(awards, "%c", list[i++]);
	fprintf(awards, "\n");

R2 :	Null
AA : DIVOR DIVO DIVO C DIVO T R2 DIVC DIVC C DIVC DIVC	Null
AA :	Null
FTS : DIVOFTS RESCON RESCON WEB AA DIVC	Null
L : LIO T LIC	Null
D : DIVO T DIVC	Null
P : PO T PC	Null
S : SO T SC	Null
H : HO T HC	Null
U : ULO T ULC	Null
A : AO T AC	Null

Rules	Action
C : COMMENTO G COMMENTC	Null

G : DIVO G
  | DIVC G
  | PO G
  | PC G
  | HO G
  | HC G
  | SO G
  | SC G
  | LIO G
  | LIC G
  | ULO G
  | ULC G
  | AO G
  | AC G
  | string G
  | FTS G
  | DN G
  | C G
  | LINK G
  | IOW G
  |
	Null

T : T D
  | T P
  | T H
  | T S
  | T U
  | T L
  | T A
  | T string
  | T FTS
  | T DN
  | T C
  | T LINK
  | T IOW
  |
	Null