FORMS

- Forms
  - For collecting user input
    - An advanced feature supported in newer versions of browsers.
    - Users may design their forms anyway to their liking.
  - Form input using <INPUT>
    - Radio button for selecting one from a set in general:

      <input type=”radio” … checked> …
      <input type=”radio” ...> …

    - Checkbox for selecting more than one:

      <input type=”checkbox” …checked> …
      <input type=”checkbox” ... > …

    - Note: Using radio buttons to select more than one is allowed if so desired.

- Form input using <SELECT>
  - Will generate a pull-down menu to select one out of many.
  - Need <option> to give values for selection.

    <select name=” … ">
    <option> …
    <option> …
    <option> …
    </select>

- Form text data input
  - For line input:

    <INPUT TYPE=”text” …>

  - For block input:

    <TEXTAREA NAME=”…” ROWS=value COLS=value> … </TEXTAREA>

- Form submission
  - Two things needed for submitting user input:
    - A submit button or an image:

      <INPUT TYPE=”submit” …>
      <INPUT TYPE=”image” SRC=”filename.gif”>

    - A program to accept the data:

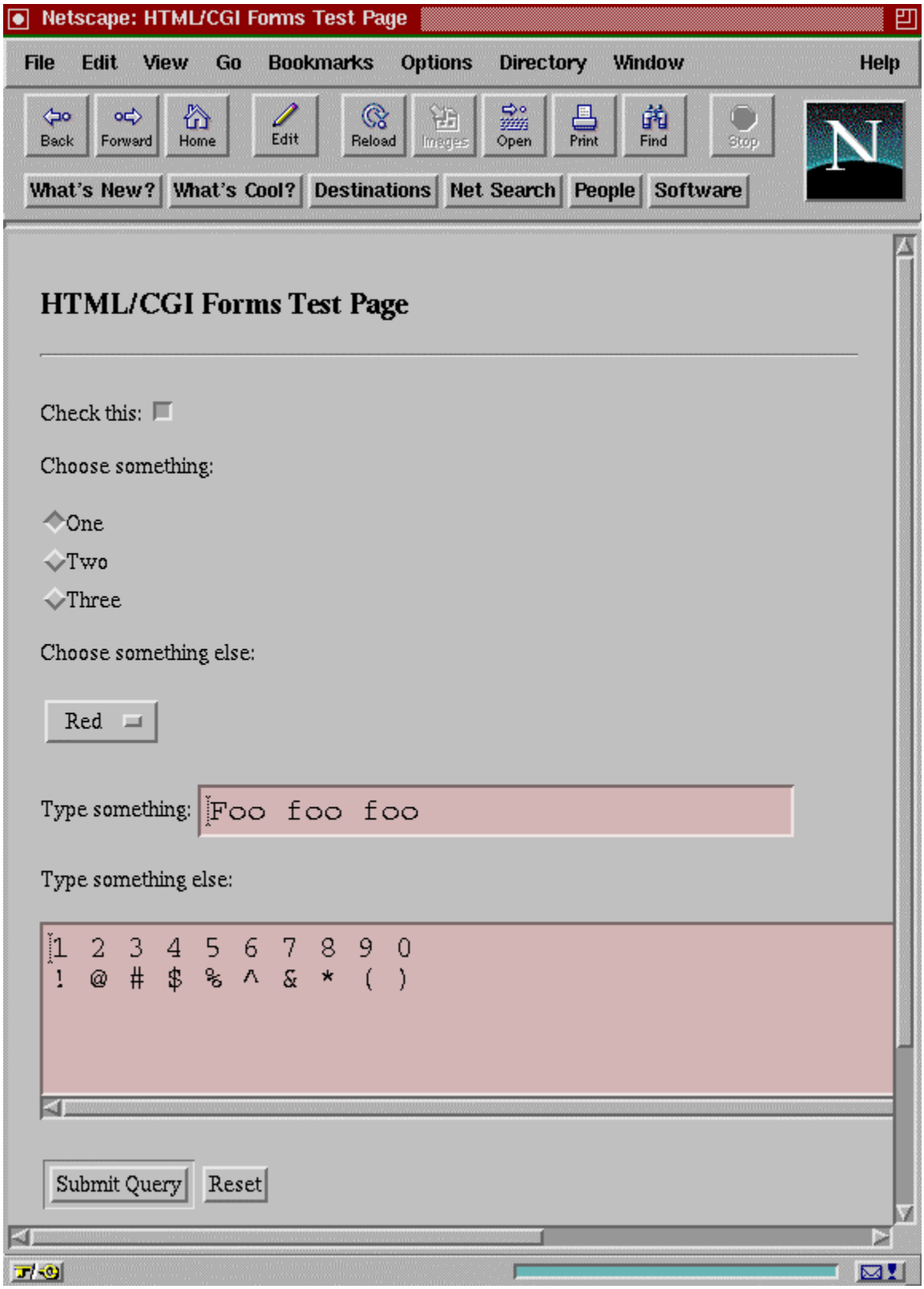
      <FORM METHOD=”post” ACTION=”filename.cgi”>

  - A reset button is nice to have.

    <INPUT TYPE=”reset”>

Example:

```
<html>
<head><title>HTML/CGI Forms Test Page</title></head>
<body>
<br>
<h2>HTML/CGI Forms Test Page</h1>
<hr><p>

<form method="post" action="http://www.cse.fau.edu/cgi-
bin/webp/form_out.cgi">

Check this: <input type="checkbox" name="check" checked><p>

Choose something:<p>

<input type="radio" name="choice1" value="One" checked>One<br>
<input type="radio" name="choice1" value="Two">Two<br>
<input type="radio" name="choice1" value="Three">Three<br><p>

Choose something else:<p>

<select name="choice2">
<option>Red
<option>Green
<option>Blue
</select><p>

Type something:
<input type="text" name="text1" size="30" value="Foo foo foo"><p>

Type something else:<p>

<textarea name="text2" rows="5" cols="70">1 2 3 4 5 6 7 8 9 0! @ # $ % ^ &
* ( )</textarea><p>

<input type="submit"> <input type="reset"><p>
</form>
</body>
</html>
```

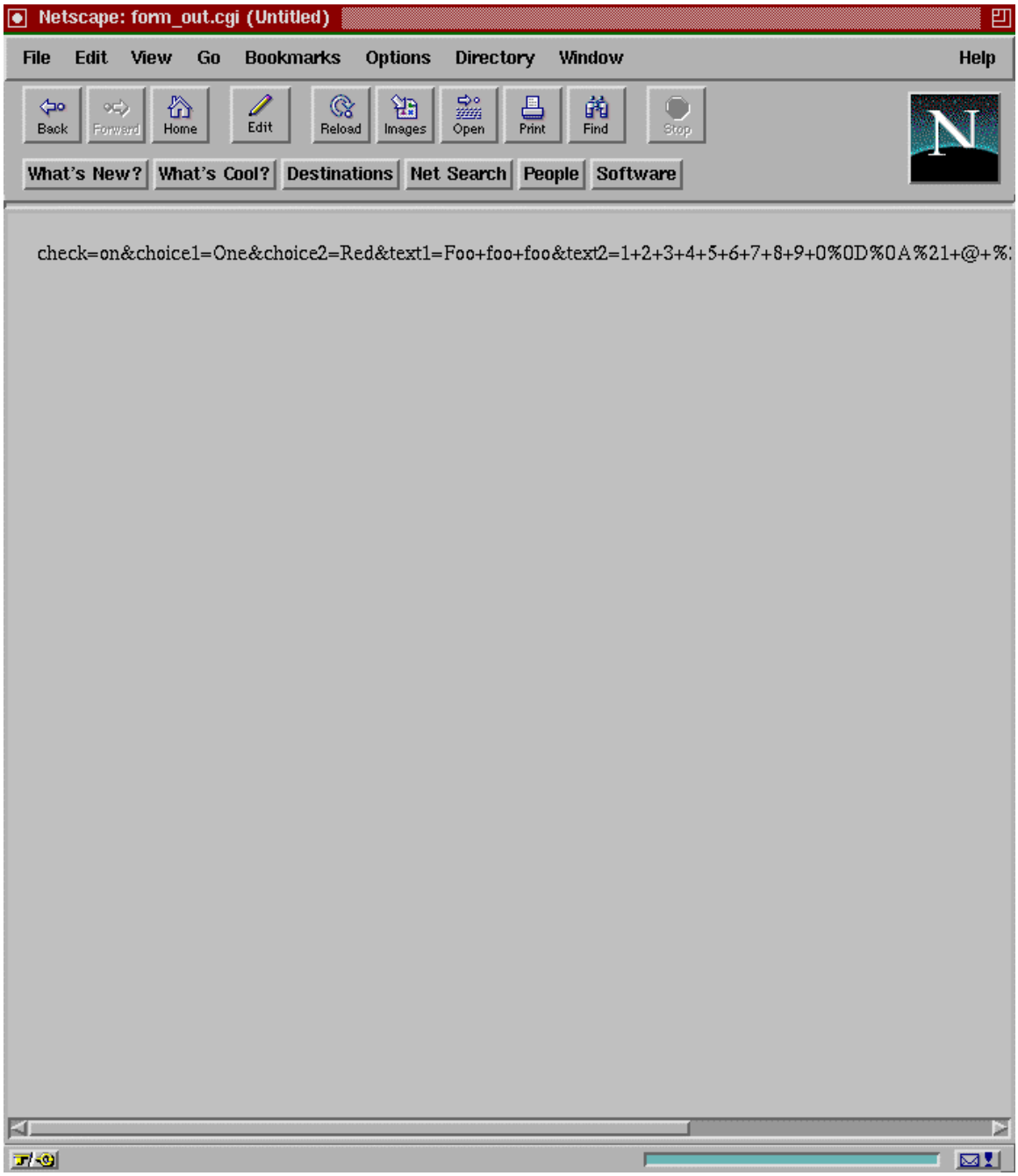

Netscape: HTML/CGI Forms Test Page
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Edit Reload Images Open Print Find Stop
What's New? What's Cool? Destinations Net Search People Software
HTML/CGI Forms Test Page
Check this:
Choose something:
One
Two
Three
Choose something else:
Red
Type something: Foo foo foo
Type something else:
1 2 3 4 5 6 7 8 9 0
! @ # $ % ^ & * ( )
Submit Query
Reset

- Form Output
  - Format of output values

    varialbe=value&variable=value& …

    where

    variable is the name defined in the form.
    value is the user input data.
    & is the delimiter.
  - Some conversions of input characters:
    - Blank character becomes **+**.
    - Special characters become **%hex**.


Netscape: form_out.cgi (Untitled)
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Edit Reload Images Open Print Find Stop
What's New? What's Cool? Destinations Net Search People Software
check=on&choice1=One&choice2=Red&text1=Foo+foo+foo&text2=1+2+3+4+5+6+7+8+9+0%0D%0A%21+@+%

- Common Gateway Interface (CGI)

  - CGI is for interaction between servers and clients.
    - Typically the client browser collects user input and submits it to the server.
    - The server, after processing the user data, returns the results to the client.
    - CGI bridges together servers and clients.
  - What is CGI?
    - A standard that specifies the data format for the client browsers and their servers to follow for data exchange.
    - CGI programming is a mechanism of implementing the communication between servers and clients.
  - CGI-bin
    - A special directory to provide CGI capabilities.
    - Typically on the server site.
    - Named ***cgi-bin*** by convention.

- Contains executable programs that may be written in any language.
  - Examples:
    - General purpose: C, C++, Java.
    - Script: Perl, UNIX Shell.
- Names end with ***.cgi*** normally.

- Is an executable program that may:
  - Gather data received from the client.
  - Send the data for processing.
  - Store the execution results in a local database, and/or returns the results to the client.

- Must be able to create an HTML document on the fly in returning the result back to the requesting client.

* HTML statement

```
<form method="post" action="http://www.cse.fau.edu/cgi-bin/webp/form_out.cgi">
```

* C source program – *form_out.c*

```
#include <stdio.h>

main()
{
  int i, len = 0;

  printf("Content-type: text/html\n\n");

  if (getenv("CONTENT_LENGTH"))   /* if this variable is not empty */
       len = atoi(getenv("CONTENT_LENGTH"));
  else
       return(-1);
  for (i = 0; i < len; i ++)
       putchar(getchar());
  putchar('\n');
  return(0);
}
```

* HTML statement

```
<form method="post" action="http://www.cse.fau.edu/cgi-bin/webp/form_parse.cgi">
```

* C source program – *form_parse.c*

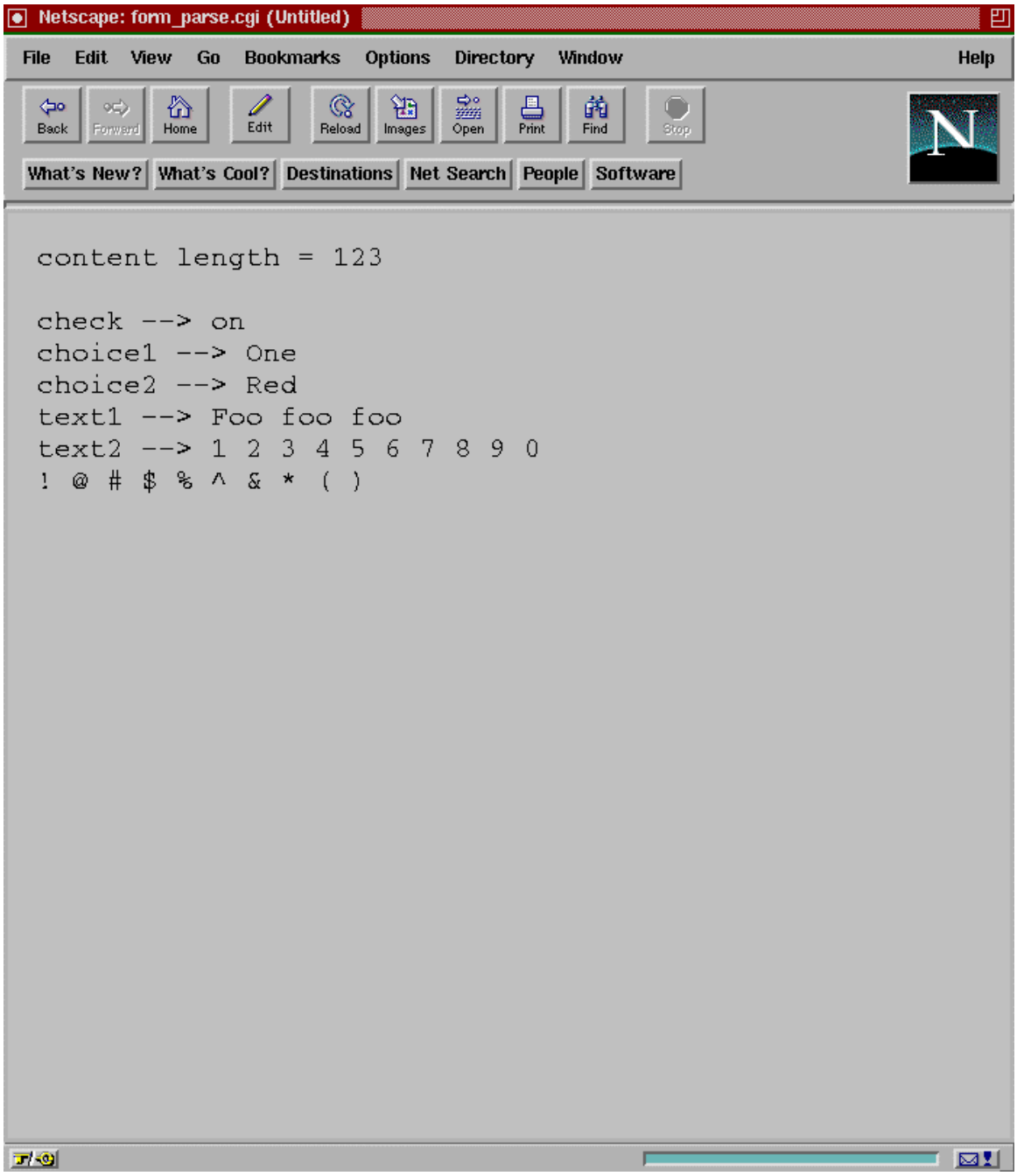
```
/* form_parse.c
   The major function of this program is to parse the data input from a
   client browser and separate variables from their values.
   The input format is in the form of variable=value&variable=value&...
*/

#include <stdio.h>
#include <stdlib.h>

enum {delim_amp = -3 , delim_eq = -2, eof = -1, val = 1, var = 2, blank = ' '};

main()
{
   int len = 0, type;
   char ch, *ptr, *get_str(int *, int *);

   printf("Content-type: text/plain\n\n");

   if (getenv("CONTENT_LENGTH")) { /* if this variable is not empty */
      len = atoi(getenv("CONTENT_LENGTH"));  /* get length in char's */
      printf("content length = %d\n\n", len);
   }
   else  {     /* failed to get anything useful */
      fprintf(stderr, "Nothing received\n");
      return(-1);          /* terminate the program */
   }

while (len) {
      ptr = get_str(&len, &type);
      if (ptr)
         printf("%s", ptr);
      if (type == var)     /* for output formatting purposes */
         printf("\t-->  ");
```

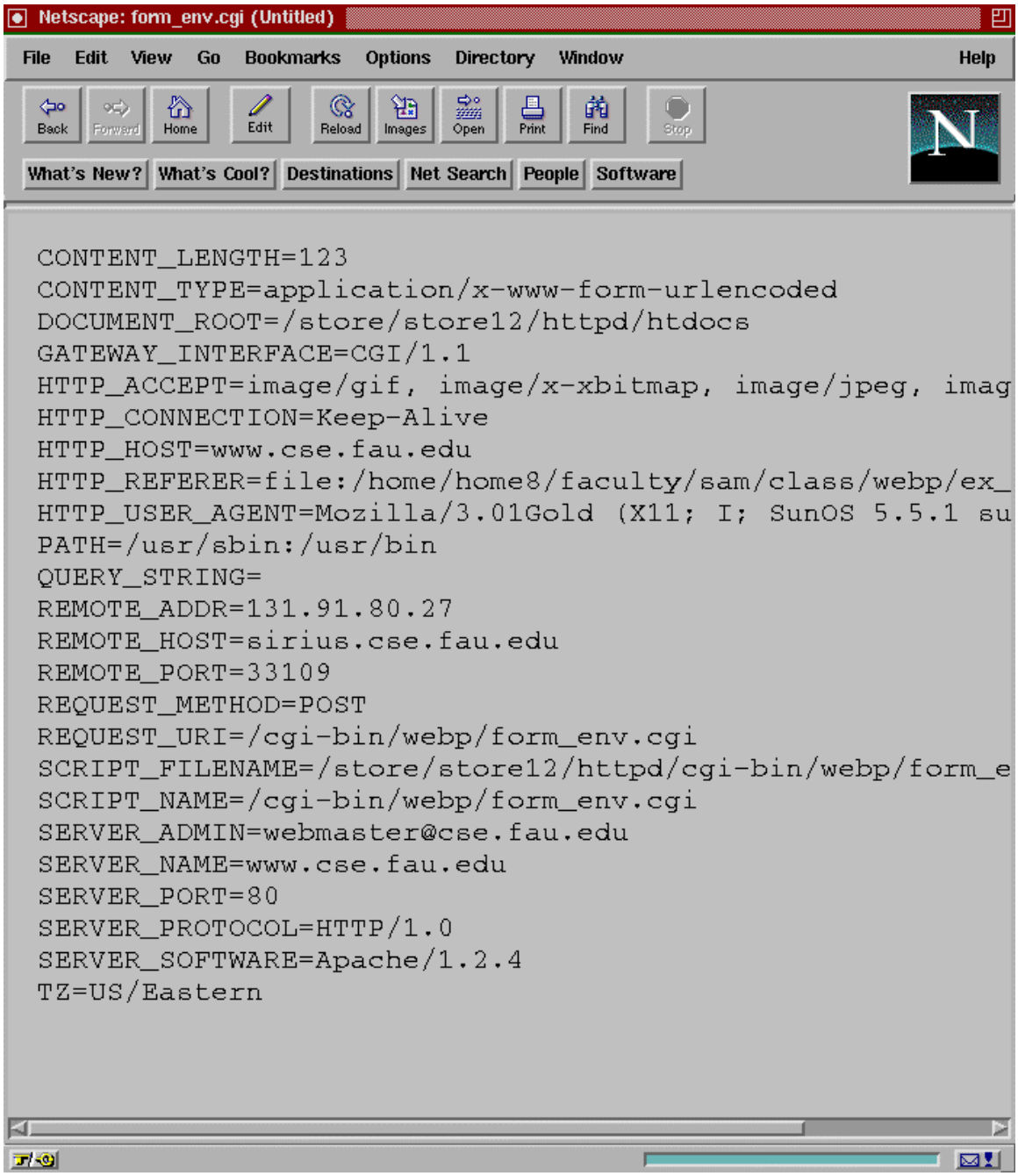
```
        else
            printf("\n");
    }
}  /* main */

/*
  get_str() is used to return a string from stdin. The return value may be
  a variable name or the character value of a variable. A NULL pointer
  is returned if nothing is encountered (e.g., an empty field, or nothing
  was selected).

  A string is terminated by a NULL character, when returned. However,
  A string is delimited by a (not include) '&', or '=' in the input data.
  Note: '&', and '=' can be a valid input character, but they are represented
  as %hh sequences in stdin.

  This function calls get_char() to get one char at a time. Space needed
  for the string to be returned will be allocated in this fucntion.

  The len parameter is used to indicate the length of the remaining input
  data in bytes, and the type parameter is used to identify the data type
  of the str to be returned: variable or value.
*/

char *get_str(int *len, int *type)
{
    char tempbuf[BUFSIZ], *ptr = (char *)NULL;
    int ch, i = 0;

    *type = val;                /* default in case of empty field */
    while (ch = get_char(len)) {
        if (ch == eof)
            break;
        tempbuf[i++] = (char)ch;
        if (ch == delim_amp) {
            i--;
            break;
        } else if (ch == delim_eq) {
            i--;
            *type = var;
            break;
        }
```

```
      else if (*len == 0)
         break;
   }
   tempbuf[i] = '\0';
   if (i) {
      ptr = malloc(i+1);
      strcpy(ptr, tempbuf);
   }
   return(ptr);
}  /* get_str */

/*
  get_char() is used to return one char at a time from stdin.
  If the char received is:
     '&':  it returns delim_amp
     '=':  it returns delim_eq
     '+':  it returns blank
     '%':  it will reconstruct the orignal special char from its hex number
  else it returns the exact char received.

  The function will also updates the len parameter depending on the number
  of char's it has read in each invocation. However, the function will return
  immediately if EOF is true.
*/

int get_char(int *len)
{
  int i = 1;
  char ch, hex[2];

  if (*len == 0)
      return(eof);          /* an empty string */

  ch = getchar();
  switch(ch) {
    case '&':               /* end of a name=value pair */
        *len -= i;          /* update length */
        return(delim_amp);
    case '=':               /* an equal character */
        *len -= i;          /* update length */
        return(delim_eq);
```

```
    case '+':               /* a blank character */
        ch = blank;
        break;
case '%':               /* a special character in a 2-digit hex */
        hex[0] = getchar();
        hex[1] = getchar();
        i = i + 2;          /* increment char counter */

      {   /* covert char hex to an int --  can be a sub-program */
        int j;
        ch = 0;             /* use ch as a small int */
        for (j = 0; j < 2; j++) {
            switch(hex[j]) {
              case '0':
                 ch = ch * 16;
                 break;
              case '1':
                 ch = ch * 16 + 1;
                 break;
              case '2':
                 ch = ch * 16 + 2;
                 break;
              case '3':
                 ch = ch * 16 + 3;
                 break;
              case '4':
                 ch = ch * 16 + 4;
                 break;
              case '5':
                 ch = ch * 16 + 5;
                 break;
              case '6':
                 ch = ch * 16 + 6;
                 break;
              case '7':
                 ch = ch * 16 + 7;
                 break;
              case '8':
                 ch = ch * 16 + 8;
                 break;
```

```
            case '9':
               ch = ch * 16 + 9;
               break;
            case 'A':
            case 'a':
               ch = ch * 16 + 10;
               break;
            case 'B':
            case 'b':
               ch = ch * 16 + 11;
               break;
            case 'C':
            case 'c':
               ch = ch * 16 + 12;
               break;
            case 'D':
            case 'd':
               ch = ch * 16 + 13;
               break;
            case 'E':
            case 'e':
               ch = ch * 16 + 14;
               break;
            case 'F':
            case 'f':
               ch = ch * 16 + 15;
               break;
          }  /* switch */
       } /* for loop */
     }   /* coversion */
     break;    /* for readability, not needed */
  }  /* outer switch */
  *len -= i;          /* update length */
  return(ch);         /* return char received or converted */
}  /* get_char */
```


Netscape: form_parse.cgi (Untitled)
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Edit Reload Images Open Print Find Stop
What's New? What's Cool? Destinations Net Search People Software
content length = 123
check --> on
choice1 --> One
choice2 --> Red
text1 --> Foo foo foo
text2 --> 1 2 3 4 5 6 7 8 9 0
! @ # $ % ^ & * ( )


Netscape: form_env.cgi (Untitled)
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Edit Reload Images Open Print Find Stop
What's New? What's Cool? Destinations Net Search People Software
CONTENT_LENGTH=123
CONTENT_TYPE=application/x-www-form-urlencoded
DOCUMENT_ROOT=/store/store12/httpd/htdocs
GATEWAY_INTERFACE=CGI/1.1
HTTP_ACCEPT=image/gif, image/x-xbitmap, image/jpeg, imag
HTTP_CONNECTION=Keep-Alive
HTTP_HOST=www.cse.fau.edu
HTTP_REFERER=file:/home/home8/faculty/sam/class/webp/ex_
HTTP_USER_AGENT=Mozilla/3.01Gold (X11; I; SunOS 5.5.1 su
PATH=/usr/sbin:/usr/bin
QUERY_STRING=
REMOTE_ADDR=131.91.80.27
REMOTE_HOST=sirius.cse.fau.edu
REMOTE_PORT=33109
REQUEST_METHOD=POST
REQUEST_URI=/cgi-bin/webp/form_env.cgi
SCRIPT_FILENAME=/store/store12/httpd/cgi-bin/webp/form_e
SCRIPT_NAME=/cgi-bin/webp/form_env.cgi
SERVER_ADMIN=webmaster@cse.fau.edu
SERVER_NAME=www.cse.fau.edu
SERVER_PORT=80
SERVER_PROTOCOL=HTTP/1.0
SERVER_SOFTWARE=Apache/1.2.4
TZ=US/Eastern

* HTML statement

```
<form method="post" action="http://www.cse.fau.edu/cgi-bin/webp/form_env.sh">
```

* Shell script – *form_env.sh*

```
#!/bin/sh
echo "Content-type: text/plain"
echo
env
```

– For more information:

http://www.efn.org/webbuild/cgiforms/cgiforms.html
http://hoohoo.ncsa.uiuc.edu/cgi/overview.html
http://www.webthing.com/tutorials/cgifaq.html
http://www.cc.ukans.edu/~acs/docs/other/forms-intro.shtml
http://snowwhite.it.brighton.ac.uk/~mas/mas/courses/html/html.html
http://wdvl.com/Authoring/Scripting/WebWare/Server