```c
// $Id: binarydump.c,v 1.3 2014-10-21 20:01:37-07 - - $

//
// Dump out files in binary.
//

#include <ctype.h>
#include <errno.h>
#include <fcntl.h>
#include <libgen.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

char *execname = NULL;
int exit_status = EXIT_SUCCESS;

void syserror (char *filename) {
   exit_status = EXIT_FAILURE;
   fflush (NULL);
   fprintf (stderr, "%s: %s: %s\n",
            execname, filename, strerror (errno));
   fflush (NULL);
}

void binary_dump (char *filename, int in_fdes) {
   printf ("%s:\n", filename);
   char buffer[8];
   ssize_t offset = 0;
   for (;;) {
      ssize_t nbytes = read (in_fdes, buffer, sizeof buffer);
      if (nbytes < 0) syserror (filename);
      if (nbytes <= 0) break;
      printf ("%7ld", offset);
      offset += nbytes;
      for (ssize_t ichar = 0; ichar < nbytes; ++ichar) {
         printf (" ");
         for (int bit = 0x80; bit != 0; bit >>= 1) {
            printf ("%d", (buffer[ichar] & bit) != 0);
         }
      }
      printf ("\n");
      printf ("%7s", "");
      for (ssize_t ichar = 0; ichar < nbytes; ++ichar) {
          char byte = buffer[ichar];
          printf (" %03o %02X", byte, byte);
          if (isgraph (byte)) printf (" %c", byte);
                         else printf ("  ");
      }
      printf ("\n");
   }
   printf ("%7ld\n", offset);
}

```

```

int main (int argc, char **argv) {
   execname = basename (argv[0]);
   if (argc == 1) {
      binary_dump ("-", STDIN_FILENO);
   }else {
      for (int argi = 1; argi < argc; ++argi) {
         char *filename = argv[argi];
         if (strcmp (filename, "-") == 0) {
            binary_dump ("-", STDIN_FILENO);
         }else {
            int in_fdes = open (filename, O_RDONLY);
            if (in_fdes < 0) {
               syserror (filename);
            }else {
               binary_dump (filename, in_fdes);
               int rc = close (in_fdes);
               if (rc < 0) syserror (filename);
            }
         }
      }
   }
   return exit_status;
}
```

```
// $Id: hexadecimaldump.c,v 1.1 2014-10-21 20:11:40-07 - - $

//
// Dump out files in binary.
//

#include <ctype.h>
#include <errno.h>
#include <fcntl.h>
#include <libgen.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

char *execname = NULL;
int exit_status = EXIT_SUCCESS;

void syserror (char *filename) {
   exit_status = EXIT_FAILURE;
   fflush (NULL);
   fprintf (stderr, "%s: %s: %s\n",
            execname, filename, strerror (errno));
   fflush (NULL);
}

void hex_dump (char *filename, int in_fdes) {
   printf ("%s:\n", filename);
   ssize_t offset = 0;
   for (;;) {
      char buffer[16];
      ssize_t nbytes = read (in_fdes, buffer, sizeof buffer);
      if (nbytes < 0) syserror (filename);
      if (nbytes == 0) break;
      printf ("%07lX", offset);
      offset += nbytes;
      for (ssize_t ichar = 0; ichar < 16; ++ichar) {
         if (ichar % 4 == 0) printf (" ");
         if (ichar < nbytes) {
            printf ("%02X", (unsigned char) buffer[ichar]);
         }else {
            printf ("  ");
         }
      }
      printf (" |");
      for (ssize_t ichar = 0; ichar < 16; ++ichar) {
          if (ichar < nbytes) {
             char byte = buffer[ichar];
             if (isprint (byte)) printf ("%c", byte);
                            else printf (".");
          }else {
             printf (" ");
          }
      }
      printf ("|\n");
   }
}
```

```c


int main (int argc, char **argv) {
   execname = basename (argv[0]);
   if (argc == 1) {
      hex_dump ("-", STDIN_FILENO);
   }else {
      for (int argi = 1; argi < argc; ++argi) {
         char *filename = argv[argi];
         if (strcmp (filename, "-") == 0) {
            hex_dump ("-", STDIN_FILENO);
         }else {
            int in_fdes = open (filename, O_RDONLY);
            if (in_fdes < 0) {
               syserror (filename);
            }else {
               hex_dump (filename, in_fdes);
               int rc = close (in_fdes);
               if (rc < 0) syserror (filename);
            }
         }
      }
   }
   return exit_status;
}

```