**alpha\_mirror: ‘H’ becomes ‘S’ (because H is the 8th letter from the start, and S is the 8th from the end).**

**Exam Question: Mirror Alphabet**

**Objective:**  
Write a program that takes a string as an argument and prints each letter in the string converted to its opposite in the alphabet. All other characters should remain unchanged.

* For example:
  + A becomes Z, B becomes Y, ..., Z becomes A.
  + a becomes z, b becomes y, ..., z becomes a.

**Requirements:**

* If the number of arguments is not 1, print a newline (\n).
* If the argument contains any non-alphabet characters, leave them unchanged.

**Examples:**

sh

Copy code

$> ./mirror\_alpha "Hello"

Svool

$> ./mirror\_alpha "42"

42

$> ./mirror\_alpha

$> (prints newline)

**#include <unistd.h>**

**int main(int argc, char \*argv[])**

**{**

**int     i;**

**char    ltr;**

**i = 0;**

**if (argc == 2)**

**{**

**while (argv[1][i])**

**{**

**ltr = argv[1][i];**

**if ('A' <= argv [1][i] && 'Z' >= argv[1][i])**

**ltr = 'Z' - argv[1][i] + 'A';**

**if ('a' <= argv[1][i] && 'z' >= argv[1][i])**

**ltr = 'z' -argv[1][i] + 'a';**

**write(1, &ltr, 1);**

**i += 1;**

**}**

**}**

**write(1, "\n", 1);**

**return (0);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Exam Question: DO\_OP (Do Operation)**

**DO­­\_OP:**

**#include <stdio.h>**

**#include <stdlib.h>**

**int main(int argc, char \*argv[])**

**{**

**if (argc == 4)**

**{**

**if (argv[2][0] == '+')**

**printf("%d", (atoi(argv[1]) + atoi(argv[3])));**

**if (argv[2][0] == '-')**

**printf("%d", (atoi(argv[1]) - atoi(argv[3])));**

**if (argv[2][0] == '\*')**

**printf("%d", (atoi(argv[1]) \* atoi(argv[3])));**

**if (argv[2][0] == '/')**

**printf("%d", (atoi(argv[1]) / atoi(argv[3])));**

**if (argv[2][0] == '%')**

**printf("%d", (atoi(argv[1]) % atoi(argv[3])));**

**}**

**printf("\n");**

**return (0);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ft\_atoi:**

**int    ft\_atoi(const char \*str)**

**{**

**int result;**

**int sign;**

**result = 0;**

**sign = 1;**

**while (\*str == 32 || (\*str >= 9 && \*str <= 13))**

**str++;**

**while (\*str == '-' || \*str == '+')**

**{**

**if (\*str == '-')**

**{**

**sign \*= -1;**

**}**

**str++;**

**}**

**while (\*str >= '0' && \*str <= '9')**

**{**

**result = result \* 10 + (\*str - '0');**

**str++;**

**}**

**return (sign \* result);**

**}**

**#include <stdio.h>**

**int main (void)**

**{**

**char \*str;**

**str = "   ---+--+12345ab67";**

**printf ("%d", ft\_atoi (str));**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ft\_strcmp:**

**int ft\_strcmp(char \*s1, char \*s2)**

**{**

**while (\*s1++ == \*s2++)**

**if (!\*s1 && !\*s2)**

**return (0);**

**return (\*--s1 - \*--s2);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ft\_strcspn:**

**#include <stdlib.h>**

**size\_t ft\_strcspn(const char \*s, const char \*reject)**

**{**

**size\_t i;**

**size\_t j;**

**size\_t len;**

**i = 0;**

**len = 0;**

**while (s[len])**

**len++;**

**while(s[i])**

**{**

**j = 0;**

**while(reject[j])**

**{**

**if (s[i] == reject[j])**

**return (i);**

**j++;**

**}**

**i++;**

**}**

**return (len);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\*ft\_strdup:**

**#include <stdlib.h>**

**char    \*ft\_strdup(char \*src)**

**{**

**int     i;**

**int     length;**

**char    \*strcpy;**

**length = 0;**

**while (src[length])**

**length++;**

**strcpy = malloc(length + 1);**

**if (strcpy != NULL)**

**{**

**i = 0;**

**while (src[i])**

**{**

**strcpy[i] = src[i];**

**i++;**

**}**

**strcpy[i] = '\0';**

**}**

**return (strcpy);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\*ft\_strrev:**

**char \*ft\_strrev(char \*str)**

**{**

**int i;**

**int len;**

**char tmp;**

**len = 0;**

**while (str[len])**

**len++;**

**i = -1;**

**while (++i < --len)**

**{**

**tmp = str[i];**

**str[i] = str[len];**

**str[len] = tmp;**

**}**

**return (str);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Iter:**

**#include <unistd.h>**

**int iter(char \*str, char c, int len)**

**{**

**int i;**

**i = 0;**

**while (str[i] && (i < len || len == -1))**

**if (str[i++] == c)**

**return (1);**

**return (0);**

**}**

**int main(int argc, char \*argv[])**

**{**

**int i;**

**if (argc == 3)**

**{**

**i = 0;**

**while (argv[1][i])**

**{**

**if (!iter(argv[1], argv[1][i], i) && iter(argv[2], argv[1][i], -1))**

**write(1, &argv[1][i], 1);**

**i += 1;**

**}**

**}**

**write(1, "\n", 1);**

**return (0);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**is\_power\_of\_2:**

**int is\_power\_of\_2(unsigned int n)**

**{**

**if (n == 0)**

**return (0);**

**else**

**return ((n & (-n)) == n ? 1: 0);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**last\_word:**

**#include <unistd.h>**

**void    last\_word(char \*str)**

**{**

**int j;**

**int i;**

**i = 0;**

**j = 0;**

**while (str[i])**

**{**

**if (str[i] == ' ' && str[i + 1] >= 33 && str[i + 1] <= 126)**

**j = i + 1;**

**i++;**

**}**

**while (str[j] >= 33 && str[j] <= 127)**

**{**

**write(1, &str[j], 1);**

**j++;**

**}**

**}**

**int     main(int argc, char \*\*argv)**

**{**

**if (argc == 2)**

**last\_word(argv[1]);**

**write(1, "\n", 1);**

**return (0);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Max:**

**int max(int \*tab, unsigned int len)**

**{**

**int max;**

**if (!len)**

**return (0);**

**max = tab[--len];**

**while (len--)**

**if (tab[len] > max)**

**max = tab[len];**

**return (max);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**print\_bits:**

**#include <unistd.h>**

**void print\_bits(unsigned char octet)**

**{**

**int i;**

**unsigned char bit;**

**i = 8;**

**while (i--)**

**{**

**bit = (octet >> i & 1) + '0';**

**write(1, &bit, 1);**

**}**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**reverse\_bits:**

**unsigned char   reverse\_bits(unsigned char octet)**

**{**

**unsigned char   res = 0;**

**int i = 8;**

**while (i > 0)**

**{**

**res = res \* 2 + (octet % 2);**

**octet = octet / 2;**

**i--;**

**}**

**return (res);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**swap\_bits:**

**unsigned char   swap\_bits(unsigned char c)**

**{**

**return ((c >> 4) | (c << 4));**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Union:**

**#include <unistd.h>**

**int     not\_seen\_before(char \*s, int max\_pos, char c)**

**{**

**int i;**

**i = -1;**

**while(++i < max\_pos)**

**if (s[i] == c)**

**return (0);**

**return (1);**

**}**

**void    ft\_union(char \*s1, char \*s2)**

**{**

**int i;**

**int j;**

**i = -1;**

**while (s1[++i])**

**if (not\_seen\_before(s1, i, s1[i]))**

**write(1, &s1[i], 1);**

**j = -1;**

**while (s2[++j])**

**if (not\_seen\_before(s1, i, s2[j]) & not\_seen\_before(s2, j, s2[j]))**

**write(1, &s2[j], 1);**

**}**

**int main(int ac, char \*\*av)**

**{**

**if (ac == 3)**

**ft\_union(av[1], av[2]);**

**write(1, "\n", 1);**

**return (0);**

**}**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**wdmatch**

**#include <unistd.h>**

**void wdmatch(char \*s1, char \*s2)**

**{**

**int len;**

**int i;**

**len = 0;**

**i = 0;**

**while (s1[len])**

**++len;**

**while (\*s2 && i < len)**

**(\*s2++ == s1[i]) ? ++i : 0;**

**if (i == len)**

**write(1, s1, len);**

**}**

**int main(int ac, char \*\*av)**

**{**

**if (ac == 3)**

**wdmatch(av[1], av[2]);**

**write(1, "\n", 1);**

**return (0);**

**}**