```
/---common.eml.h---/

/*PURPOSE*/

Provide Header for Common EML Functions

/*INCLUDES*/

<stdio.h>
<stdio.h>
<string.h>
<stddef.h>
<stdlib.h>

/*DEFINES*/

define C89 0
define C90 0
define C94 0
define C99 0
define bool
define NULL_CHAR 0

/*CHECKING COMPILER STANDARDS*/

#    if defined (__STDC__)
#        define PREDEF_STANDARD_C_1989
#        undef C89
#        define C89 1
#        if defined (__STDC_VERSION__)
#            define PREDEF_STANDARD_C_1990
#            undef C90
#            define C90 1
#            if (__STDC_VERSION__ >= 199409L)
#                define PREDEF_STANDARD_C_1994
#                undef C94
#                define C94 1
#            endif
#            if (__STDC_VERSION__ >= 199901L)
#                define PREDEF_STANDARD_C_1999
#                undef C99
#                define C99 1
#            endif
#        endif
#    endif

/*DEFINING PORTABLE BOOLEAN*/

#    if defined (PREDEF_STANDARD_C_1999) && defined (_Bool)
#         #define bool _Bool
#    else
#        define TRUE 1
#        define FALSE 0
#        define bool short int
#    endif

/*MAX AND MIN DEFINITIONS*/

#     define max(a,b)                           \
             ({                                 \
         __typeof__ (a) _a = (a);   \
                 __typeof__ (b) _b = (b);       \
                 _a > _b ? _a : _b;             \
         })


#     define min(a,b)                           \
             ({                                 \
         __typeof__ (a) _a = (a);   \
                 __typeof__ (b) _b = (b);       \
                 _a < _b ? _a : _b;             \
         })


/*FUNCTION PROTOTYPES*/

void print_eml_version ();
void print_compiler_version ();
unsigned int charlen (char* s);
void charcopy (char* cpfrom, char* cpto);
void reverse (char* s);
char* adjust (char* s);
void print_cint (char* s);
char* print_cint_to_str (char* s);
void addchar1 (char* a, char* sum, char* dig_sum);
void addchar2 (char* a, char* b, char* sum, char* dig_sum);
char* max_cint (char* a, char* b);
char* min_cint (char* a, char* b);
char* add_cint (char* a, char* b);


/*EML Functions*/

void print_eml_version ()
{
    printf("%s\n", EML_VERSION);
}

void print_compiler_version ()
{
    if (C99 == 1)
    {
    printf("%s\n", "C99");
    }
    else if (C94 == 1)
    {
    printf("%s\n", "C94");
    }
    else if (C90 == 1)
    {
    printf("%s\n", "C94");
    }
    else if (C89 == 1)
    {
    printf("%s\n", "C89");
    }
    else
    {
    printf("%s\n", "Unknown Version");
    }
}

/*Char functions*/
```

```
unsigned int charlen (char* s)
{
    unsigned int len = 0;

    while (*(s+len) != '\0')
    {
    len++;
    }

    return len;
}

void charcopy (char* cpfrom, char* cpto)
{
    while ((*cpto = *cpfrom) != '\0')
    {
        cpto++;
        cpfrom++;
    }
}


void reverse (char* s)
{
    unsigned int len = charlen (s);
    char temp;

    for(int var = 0; var < len / 2; var++)
    {
    temp = *(s+var);
    *(s+var) = *(s+len-1-var);
    *(s+len-1-var) = temp;
    }
}

char* adjust (char* s)
{
    char* c;
    c = (char*) malloc (charlen (s) * sizeof(char));
    charcopy (s, c);
    return c;
}

void print_cint (char* s)
{
    unsigned int len =  charlen (s);

    while (len > 0)
    {
    putchar (*(s+len-1));
    len--;
    }
}


char* print_cint_to_str (char* s)
{
    unsigned int len = charlen (s);
    char* c = (char*) malloc (len * sizeof (char));
    while (len > 0)
    {
        *c = *(s+len-1);
        c++;
        len--;
    }
    return c;
}

/*Operations on char int or C-int*/

void addchar1 (char* a, char* sum, char* dig_sum)
{
    *sum = (*a - '0') + (*dig_sum);
    *dig_sum = *sum / 10;
    *sum = (*sum % 10) + '0';
}

void addchar2 (char* a, char* b, char* sum, char* dig_sum)
{
    *sum = (*a - '0') + (*b - '0') + (*dig_sum);
    *dig_sum = *sum / 10;
    *sum = (*sum % 10) + '0';
}

char* max_cint (char* a, char* b)
{
    unsigned int len_a = charlen (a);
    unsigned int len_b = charlen (b);

    if (len_a > len_b)
    {
    return a;
    }
    else if (len_a < len_b)
    {
    return b;
    }
    else if (*(a+len_a-1) > *(b+len_b-1))
    {
    return a;
    }
    else
    {
    return b;
    }
}

char* min_cint (char* a, char* b)
{
    unsigned int len_a = charlen (a);
    unsigned int len_b = charlen (b);

    if (len_a < len_b)
    {
    return a;
    }
    else if (len_a > len_b)
    {
    return b;
    }
    else if (*(a+len_a-1) < *(b+len_b-1))
    {
    return a;
    }
    else
    {
    return b;
    }
}

char* add_cint (char* a, char* b)
{
    unsigned int len_a = charlen (a);
    unsigned int len_b = charlen (b);

    unsigned int min_len = min (len_a, len_b);
    unsigned int max_len = max (len_a, len_b);

    unsigned int sum_len = max_len + 1;

    char* sum = (char*) malloc (sum_len * sizeof (char));

    char* digit_sum = (char*) malloc (sizeof (char));
    *digit_sum = 0;

    a += len_a-1;
    b += len_b-1;

    while (min_len > 0)
    {
    addchar2 (a--, b--, sum++, digit_sum);
    min_len--;
    }

    sum_len -= 1;

    if (len_a >= len_b)
    {
    while (sum_len > len_b)
    {
        addchar1 (a--, sum++, digit_sum);
        sum_len--;
    }
    }
    else
    {
    while (sum_len > len_a)
    {
        addchar1 (b--, sum++, digit_sum);
        sum_len--;
    }
    }

    if (*digit_sum > 0)
    {
    *sum = *digit_sum + '0';
    sum++;
    *sum = '\0';

    return (sum-max_len-1);

    }

    *sum = '\0';

    return adjust(sum-max_len);
```

```
}
```

//----------End----------//