

```
1 // Created by hfwei on 2024/10/30.
2
3 #include <limits.h>
4 #include <stdio.h>
5
6 int main() {
7     printf("CHAR_MIN = %d\n", CHAR_MIN);
8     printf("CHAR_MAX = %d\n", CHAR_MAX);
9
10    char c = 150;
11    int i = 900;
12
13    printf("i / c = %d\n", i / c);
14
15    return 0;
16 }
```

```
1 // Created by hengxin on 2024/10/30.
2
3 #include <stdio.h>
4
5 int main() {
6     /**
7      * Do not use a counter of type float/double,
8      * although it works on some platforms.
9      *
10     * 0.1 cannot be exactly represented in machines.
11     */
12     for (double x = 0.1; x <= 1.0; x += 0.1) {
13         printf("%.20f\n", x);
14     }
15
16     return 0;
17 }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <stdio.h>
4
5 int main() {
6     // Integer types
7     printf("Size of char: %zu bytes\n", sizeof(char));
8     printf("Size of signed char: %zu bytes\n", sizeof(signed
9     char));
10    printf("Size of unsigned char: %zu bytes\n\n", sizeof(
11    unsigned char));
12    printf("Size of short: %zu bytes\n", sizeof(short));
13    printf("Size of unsigned short: %zu bytes\n\n", sizeof(
14    unsigned short));
15    printf("Size of int: %zu bytes\n", sizeof(int));
16    printf("Size of unsigned int: %zu bytes\n\n", sizeof(
17    unsigned int));
18    printf("Size of long: %zu bytes\n", sizeof(long));
19    printf("Size of unsigned long: %zu bytes\n\n", sizeof(
20    unsigned long));
21    printf("Size of long long: %zu bytes\n", sizeof(long
22    long));
23    printf("Size of unsigned long long: %zu bytes\n\n",
24    sizeof(unsigned long long));
25    // (Real) Floating-point types
26    printf("Size of float: %zu bytes\n", sizeof(float));
27    printf("Size of double: %zu bytes\n", sizeof(double));
28    printf("Size of long double: %zu bytes\n\n", sizeof(long
29    double));
30    // Array types
31    int numbers[] = {0, 1, 2, 3, 4};
32    size_t len = sizeof numbers / sizeof(int);
33    printf("Length of numbers: %zu\n", len);
34    return 0;
35 }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <limits.h>
4 #include <stdio.h>
5
6 #define SIZE UINT_MAX
7
8 char string[SIZE] = {'A', 'B', 'C', 'D', 'E', 'F'};
9
10 void Print(const int string[], size_t size);
11
12 int main(void) {
13     Print(string, SIZE);
14
15     return 0;
16 }
17
18 void Print(const int string[], size_t size) {
19     for (int i = 0; i < size; i++) {
20         printf("%d : %d\n", i, string[i]);
21     }
22 }
```

```
1 // Created by hfwei on 2024/10/30.
2
3 #include <stdio.h>
4 #include <time.h>
5
6 long long Fib(int n);
7
8 int main() {
9     int n;
10    scanf("%d", &n);
11
12    time_t start = time(NULL);
13    printf("Fib(%d) = %lld\n", n, Fib(n));
14    time_t end = time(NULL);
15
16    return 0;
17 }
18
19 long long Fib(int n) {
20     if (n <= 1) {
21         return n;
22     }
23
24     return Fib(n - 1) + Fib(n - 2);
25 }
```

```
1  /**
2   * See
3   * https://randomascii.wordpress.com/2012/02/25/comparing-
   floating-point-numbers-2012-edition/
4   *
5   * Created by hfwei on 2024/10/31.
6   */
7
8  #include <float.h>
9  #include <math.h>
10 #include <stdbool.h>
11 #include <stdio.h>
12
13 #define EPSILON 1e-5
14
15 bool IsEqual(double x, double y);
16
17 int main() {
18     printf("%d\n", IsEqual(DBL_MAX, DBL_MAX - 100));
19     printf("%d\n", !islessgreater(DBL_MAX, DBL_MAX - 100));
20
21     printf("%.50f\n", DBL_MAX - (DBL_MAX - 100));
22     printf("%d\n", !islessgreater(DBL_MAX, DBL_MAX - 100));
23
24     return 0;
25 }
26
27 bool IsEqual(double x, double y) { return fabs(x - y) <=
    EPSILON; }
```

```
1 # 7-data-types
2
3 ## `int-limits.c`
4
5 ## `unsigned.c`
6
7 ## `timing.c`
8
9 ## `char.c`
10
11 ## `int-overflow.c`
12
13 ## `implicit-inversion.c`
14
15 ## `explicit-inversion.c`
16
17 ## `float-limits.c`
18
19 ## `sums.c`
20
21 ## `loop.c`
22
23 ## `compare.c`
```

```
1 // Created by hfwei on 2024/10/10.
2
3 #include <stdio.h>
4
5 int main() {
6     const int array[] = {0, 1, 2, 3, 4};
7     int i = -1;
8
9     size_t size = sizeof array / sizeof array[0];
10    printf("The size of the array is %zu\n", size);
11
12    if (i <= size) {
13        printf("i <= sizeof array\n");
14    } else {
15        printf("i > sizeof array\n");
16    }
17
18    return 0;
19 }
```



```
1 // Created by hfwei on 2024/10/31.
2
3 #include <limits.h>
4 #include <stdio.h>
5 #include <stdlib.h>
6
7 unsigned int pow2(unsigned int exp);
8
9 int main(void) {
10     unsigned int exp = 30;
11
12     unsigned int pow = pow2(exp);
13     printf("2^%d = %d\n", exp, pow);
14
15     return 0;
16 }
17
18 unsigned int pow2(unsigned int exp) {
19     if (exp >= sizeof(unsigned int) * CHAR_BIT) {
20         printf("Exp is too large!\n");
21         exit(1);
22     }
23
24     return 1 << exp;
25 }
```

```
1 // Created by hfwei on 2024/10/30.
2 // Run on Windows and Linux
3
4 #include <limits.h>
5 #include <stdio.h>
6
7 int main() {
8     // INT_MIN = -2147483648
9     // INT_MAX = 2147483647 (10 digits)
10    printf("INT_MIN = %d\n", INT_MIN);
11    printf("INT_MAX = %d\n\n", INT_MAX);
12
13    // printf("UINT_MIN = %u\n", 0U);
14    // printf("UINT_MAX = %u\n\n", UINT_MAX);
15
16    printf("LONG_MIN = %ld\n", LONG_MIN);
17    printf("LONG_MAX = %ld\n\n", LONG_MAX);
18
19    // printf("ULONG_MIN = %lu\n", 0UL);
20    // printf("ULONG_MAX = %lu\n\n", ULONG_MAX);
21
22    // long long int: >= 64 bits
23
24    // LLONG_MIN = -9223372036854775808
25    // LLONG_MAX = 9223372036854775807 (19 digits)
26    printf("LLONG_MIN = %lld\n", LLONG_MIN);
27    printf("LLONG_MAX = %lld\n\n", LLONG_MAX);
28
29    // printf("ULONG_LONG_MIN = %llu\n", 0ULL);
30    // printf("ULONG_LONG_MAX = %llu\n\n", ULONG_LONG_MAX);
31    //
32    // printf("ULLONG_MAX = %llu\n\n", ULLONG_MAX);
33
34    return 0;
35 }
```

```
1 // Created by hfwei on 2024/10/31.  
2  
3 #include <stdint.h>  
4  
5 int main(void) {  
6     int8_t small = -100;  
7     int32_t large = 100000;  
8  
9     return 0;  
10 }
```

```
1 /**
2  * file: sums.c
3  * See
4  * https://randomascii.wordpress.com/2012/02/25/comparing-
5  \* floating-point-numbers-2012-edition/
6  * Created by hengxin on 2024/10/30.
7  */
8
9 #include <stdio.h>
10
11 int main() {
12     // 0.1: 0.0 0011 0011 0011
13     float f = 0.1F;
14
15     float sum = 0.0F;
16     for (int i = 0; i < 10; ++i) {
17         sum += f;
18     }
19
20     float product = f * 10;
21
22     printf("sum = %.15f\nmul = %.30f\n", sum, product);
23
24     return 0;
25 }
```

```
1 # Objects, size, precision, width, limits
2 add_executable(size size.c)
3 add_executable(precision precision.c)
4 add_executable(int-limits int-limits.c)
5 add_executable(exact-width exact-width.c)
6
7 add_executable(unsigned unsigned.c)
8 add_executable(sizet sizet.c)
9 add_executable(timing-primes timing-primes.c)
10
11 add_executable(char char.c)
12
13 add_executable(unsinged-wrap unsigned-wrap.c)
14 add_executable(for-unsigned for-unsigned.c)
15 add_executable(unsigned-wrap-fix unsigned-wrap-fix.c)
16
17 add_executable(signed-overflow-fix signed-overflow-fix.c)
18
19 add_executable(implicit-conversion implicit-conversion.c)
20 add_executable(integer-promotion integer-promotion.c)
21 add_executable(explct-conversion explct-conversion.c)
22
23 add_executable(float-limits float-limits.c)
24
25 add_executable(sum-product sum-product.c)
26 add_executable(loop loop.c)
27
28 add_executable(compare compare.c)
29 target_link_libraries(compare m)
```

```
1 // Created by hfwei on 2024/10/30.
2
3 #include <float.h>
4 #include <stdio.h>
5
6 int main() {
7     // float pi = 3.14F;
8
9     // 3.402823e+38
10    printf("FLT_MAX = %e\n", FLT_MAX);
11    // 1.175494e-38
12    printf("FLT_MIN = %e\n", FLT_MIN);
13    // 1.401298e-45
14    printf("FLT_TRUE_MIN = %e\n", FLT_TRUE_MIN);
15    // 1.192093e-07
16    printf("FLT_EPSILON = %e\n\n", FLT_EPSILON);
17
18    // %lf for scanf
19    // 1.797693e+308
20    printf("DBL_MAX = %e\n", DBL_MAX);
21    // 2.225074e-308
22    printf("DBL_MIN = %e\n", DBL_MIN);
23    // 4.940656e-324
24    printf("DBL_TRUE_MIN = %e\n", DBL_TRUE_MIN);
25    // 2.220446e-16
26    printf("DBL_EPSILON = %e\n\n", DBL_EPSILON);
27
28    return 0;
29 }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <stdio.h>
4 #define LEN 100
5
6 int main(void) {
7     int numbers[LEN] = {0};
8
9     for (unsigned int i = LEN; i >= 0; i--) {
10         printf("%u : %d\n", i, numbers[i]);
11     }
12
13     return 0;
14 }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <stdbool.h>
4 #include <stdio.h>
5 #include <time.h>
6
7 bool IsPrime(int number);
8
9 int main(void) {
10     int max = 0;
11     scanf("%d", &max);
12
13     int count = 0;
14
15     // return the current time in seconds since the Unix
    epoch (January 1, 1970)
16     time_t start = time(NULL);
17     for (int number = 2; number <= max; number++) {
18         if (IsPrime(number)) {
19             count++;
20         }
21     }
22     printf("\ncount = %d\n", count);
23
24     // return the current time in seconds since the Unix
    epoch (January 1, 1970)
25     time_t end = time(NULL);
26     printf("Time elapsed: %lld seconds\n", end - start);
27
28     return 0;
29 }
30
31 bool IsPrime(int number) {
32     for (int factor = 2; factor * factor <= number; factor
    ++){
33         if (number % factor == 0) {
34             return false;
35         }
36     }
37
38     return true;
39 }
```



```
1 // Created by hfwei on 2024/10/30.
2
3 #include <limits.h>
4 #include <stdio.h>
5
6 int main() {
7     printf("UINT_MAX = %u\n", UINT_MAX);
8
9     unsigned int max = UINT_MAX;
10    unsigned int one = 1U;
11    unsigned int two = 2U;
12
13    printf("max + one = %u\n", max + one);
14    printf("one - two = %u\n", one - two);
15
16    return 0;
17 }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <stdio.h>
4
5 int main(void) {
6     signed char left = 100;
7     signed char mid = 3;
8     signed char right = 4;
9
10    signed char result = left * mid / right;
11
12    printf("result = %d\n", result);
13
14    return 0;
15 }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <limits.h>
4 #include <stdio.h>
5 #include <stdlib.h>
6
7 unsigned int Add(unsigned int left, unsigned int right);
8 unsigned int Sub(unsigned int left, unsigned int right);
9 unsigned int Mul(unsigned int left, unsigned int right);
10 unsigned int Div(unsigned int left, unsigned int right);
11 unsigned int Mod(unsigned int left, unsigned int right);
12
13 int main(void) {
14     // addition
15     unsigned int left_add = UINT_MAX / 2 + 1;
16     unsigned int right_add = UINT_MAX / 2 + 1;
17
18     printf("%u + %u = %u\n\n", left_add, right_add, Add(
19         left_add, right_add));
20
21     // subtraction
22     unsigned int left_sub = 1;
23     unsigned int right_sub = 2;
24
25     printf("%u - %u = %u\n\n", left_sub, right_sub, Sub(
26         left_sub, right_sub));
27
28     // multiplication
29     unsigned int left_mul = UINT_MAX;
30     unsigned int right_mul = 2;
31
32     printf("%u * %u = %u\n", left_mul, right_mul, Mul(
33         left_mul, right_mul));
34
35     // division
36     unsigned int left_div = 5;
37     unsigned int right_div = 0;
38
39     printf("%u * %u = %u\n", left_div, right_div, Div(
40         left_div, right_div));
41     printf("%u * %u = %u\n", left_div, right_div, Mod(
42         left_div, right_div));
43
44     return 0;
45 }
```

```
40 }
41
42 unsigned int Add(unsigned int left, unsigned int right) {
43     // return left + right;
44
45     if (left > UINT_MAX - right) {
46         printf("Too Big!\n");
47         exit(1);
48     } else {
49         unsigned int sum = left + right;
50         return sum;
51     }
52 }
53
54 unsigned int Sub(unsigned int left, unsigned int right) {
55     // return left - right;
56
57     if (left < right) {
58         printf("The result is negative!\n");
59         exit(1);
60     } else {
61         unsigned int sub = left - right;
62         return sub;
63     }
64 }
65
66 unsigned int Mul(unsigned int left, unsigned int right) {
67     // return left * right;
68
69     if (left > UINT_MAX / right) {
70         printf("The result is negative!\n");
71         exit(1);
72     } else {
73         unsigned int mul = left * right;
74         return mul;
75     }
76 }
77
78 unsigned int Div(unsigned int left, unsigned int right) {
79     if (right == 0) {
80         printf("Division by zero!\n");
81         exit(1);
82     }
83 }
```

```
84     return left / right;
85 }
86
87 unsigned int Mod(unsigned int left, unsigned int right
    ) {
88     if (right == 0) {
89         printf("Division by zero!\n");
90         exit(1);
91     }
92
93     return left % right;
94 }
```

```
1 // Created by hfwei on 2024/10/30.
2
3 #include <limits.h>
4 #include <stdio.h>
5
6 int main() {
7     double pi = 3.14159;
8
9     // below: obtain its fractional part
10    double fraction = pi - (int)pi;
11
12    int num = 100000000; // (9 digits)
13    printf("LLONG_MAX = %lld\n", LLONG_MAX);
14    long long llint = (long long)num * num;
15    printf("i = %lld\n", llint);
16
17    return 0;
18 }
```

```
1 // Created by hfwei on 2024/10/30.
2
3 #include <limits.h>
4 #include <stdio.h>
5
6 int SquareInt(int num);
7 double SquareDouble(double num);
8
9 int main() {
10     // narrowing conversion (still in the range)
11     int i = 3.14159;
12
13     // out of the range: undefined behavior!!!
14     int j = UINT_MAX;
15
16     // arguments; narrowing conversion
17     double pi = 3.14;
18     SquareInt(pi);
19
20     // return value; narrowing conversion
21     int val = SquareDouble(pi);
22
23     // from int to float; narrowing conversion
24     int big = 1234567890;
25     float approx = big;
26
27     printf("big = %d\t approx = %f\t diff = %d\n", big,
28           approx,
29           big - (int)approx);
30     return 0;
31 }
32
33 int SquareInt(int num) { return num * num; }
34
35 double SquareDouble(double num) { return num * num; }
```

```
1 // Created by hfwei on 2024/10/31.
2
3 #include <limits.h>
4 #include <stdio.h>
5 #include <stdlib.h>
6
7 int Add(int left, int right);
8 int Sub(int left, int right);
9 int Mul(int left, int right);
10 int Div(int left, int right);
11 int Mod(int left, int right);
12 int Neg(int left);
13
14 int main(void) {
15     // addition
16     int left_add = INT_MAX / 2 + 1;
17     int right_add = INT_MAX / 2 + 1;
18
19     printf("%d + %d = %d\n\n", left_add, right_add, Add(
        left_add, right_add));
20
21     // subtraction
22     int left_sub = INT_MIN;
23     int right_sub = 1;
24
25     printf("%d - %d = %d\n\n", left_sub, right_sub, Sub(
        left_sub, right_sub));
26
27     // multiplication
28     int left_mul = INT_MAX;
29     int right_mul = 2;
30
31     printf("%d * %d = %d\n", left_mul, right_mul, Mul(
        left_mul, right_mul));
32
33     // division
34     int left_div = INT_MIN;
35     int right_div = -1;
36
37     printf("%d / %d = %d\n", left_div, right_div, Div(
        left_div, right_div));
38
39     // mod (remainder)
40     int left_mod = INT_MIN;
```



```
41  int right_mod = -1;
42
43  printf("%d %% %d = %d\n", left_mod, right_mod, Mod(
    left_mod, right_mod));
44
45  // negation
46  int left_neg = INT_MIN;
47
48  printf("-%d = %d\n", left_neg, Neg(left_neg));
49
50  return 0;
51 }
52
53 int Add(int left, int right) {
54     // int sum = left + right;
55     // return sum;
56
57     // if (left + right > INT_MAX) {
58     //     printf("Too Big!\n");
59     //     exit(1);
60     // } else {
61     //     int sum = left + right;
62     //     return sum;
63     // }
64
65     if ((left > 0 && right > INT_MAX - left) ||
66         (left < 0 && right < INT_MIN - left)) {
67         printf("Overflow!\n");
68         exit(1);
69     } else {
70         int sum = left + right;
71         return sum;
72     }
73 }
74
75 int Sub(int left, int right) {
76     // int sub = left - right;
77     // return sub;
78
79     // if (left - right < 0) {
80     //     printf("The result is negative!\n");
81     //     exit(1);
82     // } else {
83     //     int sub = left - right;
```

```
84     //     return sub;
85     // }
86
87     if ((left > 0 && right < INT_MIN + left) ||
88         (left < 0 && right > INT_MAX + left)) {
89         printf("Overflow!\n");
90         exit(1);
91     } else {
92         int sub = left - right;
93         return sub;
94     }
95 }
96
97 int Mul(int left, int right) {
98     // int mul = left * right;
99     // return mul;
100
101     // if (left * right > INT_MAX) {
102     //     printf("The result is negative!\n");
103     //     exit(1);
104     // } else {
105     //     int mul = left * right;
106     //     return mul;
107     // }
108
109     if (left > 0) {
110         if (right > 0) { // left > 0 && right > 0
111             if (left > INT_MAX / right) {
112                 printf("Overflow!\n");
113                 exit(1);
114             }
115         } else { // left > 0 && right < 0
116             if (right < INT_MIN / left) {
117                 printf("Overflow!\n");
118                 exit(1);
119             }
120         }
121     } else { // left <= 0
122         if (right > 0) { // left <= 0 && right > 0
123             if (left < INT_MIN / right) {
124                 printf("Overflow!\n");
125                 exit(1);
126             }
127         } else { // left <= 0 && right <= 0
```

```
128     if (left != 0 && right < INT_MAX / left) {
129         printf("Overflow!\n");
130         exit(1);
131     }
132 }
133 }
134
135 int mul = left * right;
136 return mul;
137 }
138
139 int Div(int left, int right) {
140     if (right == 0 || (left == INT_MIN && right == -1)) {
141         printf("Overflow!\n");
142         exit(1);
143     }
144
145     return left / right;
146 }
147
148 int Mod(int left, int right) {
149     if (right == 0 || (left == INT_MIN && right == -1)) {
150         printf("Overflow!\n");
151         exit(1);
152     }
153
154     return left % right;
155 }
156
157 int Neg(int left) {
158     if (left == INT_MIN) {
159         printf("Overflow!\n");
160         exit(1);
161     }
162
163     return -left;
164 }
```