SUSTech Southern University of Science and Technology

CS111，C Programming

Lab / Function

黄嘉炜

huangjw3@mail.sustech.edu.cn

深港微电子学院
SCHOOL OF MICROELECTRONICS

Outline

SUSTech Southern University of Science and Technology

- Review
- Function: Basic Intro
- Assignment

Review：remove char / timeout again!

SUSTech Southern University of Science and Technology

#5	✓ Accepted	6	4ms	324 KiB
#6	✕ Time Exceeded	0	≥1100ms	≥1 MiB
#7	✕ Time Exceeded	0	≥1100ms	≥1.4 MiB
#8	✕ Time Exceeded	0	≥1101ms	≥1.8 MiB
#9	✕ Time Exceeded	0	≥1101ms	≥1.9 MiB
#10	✕ Time Exceeded	0	≥1101ms	≥2.1 MiB
#11	✕ Time Exceeded	0	≥1101ms	≥2.2 MiB
#12	✕ Time Exceeded	0	≥1100ms	≥2.1 MiB
#13	✕ Time Exceeded	0	≥1101ms	≥2.1 MiB

Review：Search keywords

```
13      char* a=str;
14      int NNN=0;
15      for(;*a!='\0'&&NNN<top_n;++a)
16      {
17          if((*a==*keyword)|| //hit 1st keyword char
18          {
19              char*b=keyword;
20              for(int x=0;*b!='\0';++b,++x)
21              {
22                  ...(
23
24                  // break if {}, when not matched
25
26              }
27              NNN+=1;
28              *positions=a-str;
29              positions+=1;
30          }
31          qwert:;
32      }
33      return NNN;
```

Meaning?

Review：Search keywords / Extended

SUSTech Southern University of Science and Technology

```
3   void vector_pointwise_square(int* pval_start, int* pval_end)
4   {
5       for (int* p = pval_start; p <= pval_end; p++) {
6           int idx = p - pval_start;
7           *p = (*p) * (*p);
8           printf("vector_pointwise_square, idx %d, result %d \n", idx, *p);
9       }
10  }
```

offset of memory?

When p = 0x04,

pval_start = 0x00

idx = ?

Review：Extract array from string

SUSTech Southern University of Science and Technology

How to parse integer value from string?

Given: char* str = “123”;

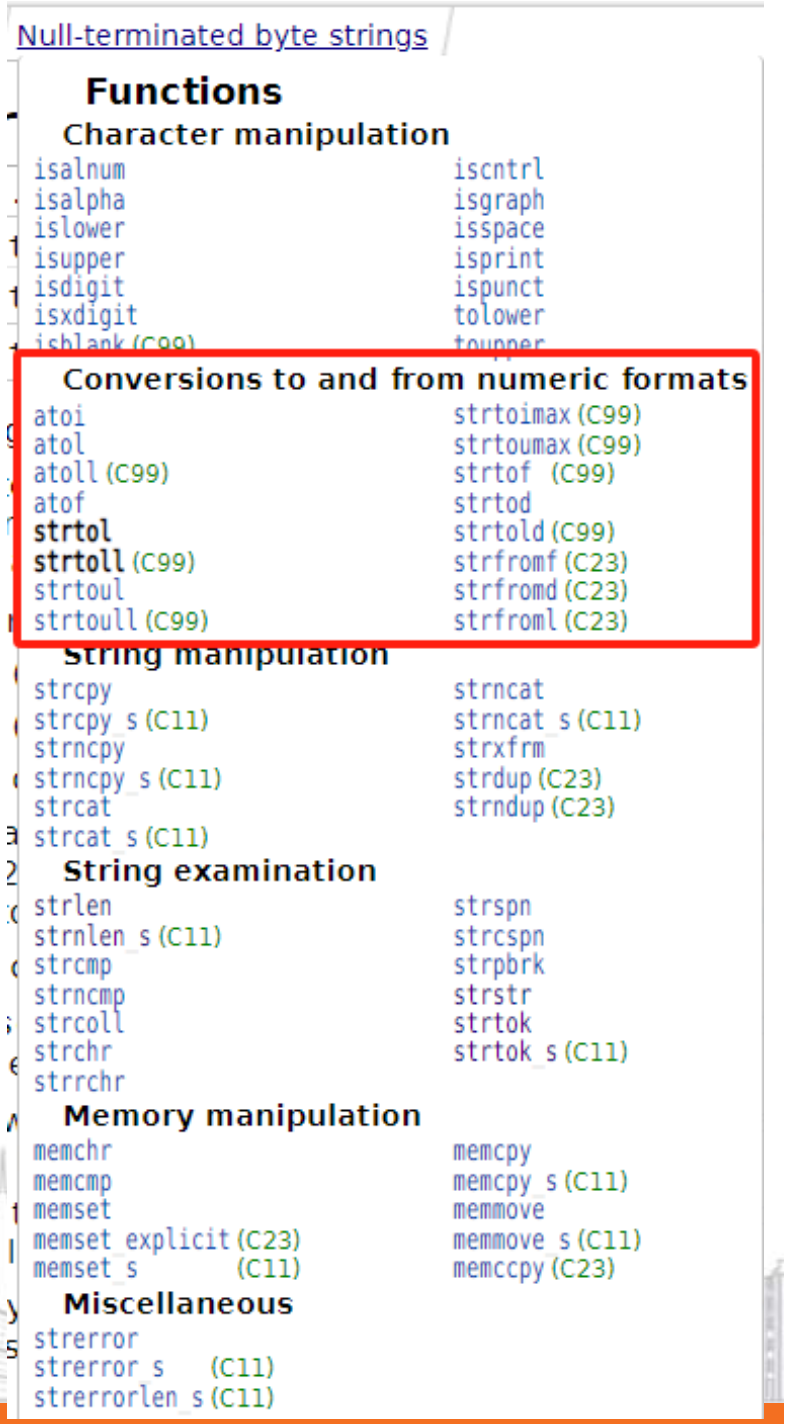
Review： Extract array from string

How to parse integer value from string?

Given: char* str = “123”；

2 ways

- **Coding my myself**
- **Call the function from library.**



Outline

SUSTech Southern University of Science and Technology

- Review
- **Function: Basic Intro**
- Assignment

Function: Basic Intro

SUSTech Southern University of Science and Technology

```c
#include <stdio.h>

int main()
{
    int values[] = {1, 2, 3, 4, 5};
    int result = sum(values, sizeof(values) / sizeof(int));
    printf("sum of values: %d \n", result);
    return 0;
}

int sum(int* values, int size)
{
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += values[i];
    }
    return sum;
}
```

Bug ?

函数的调用

函数的声明和实现

Function: Basic Intro

```
1   #include <stdio.h>
2
3   int sum(int* values, int size)
4   {
5       int sum = 0;
6       for (int i = 0; i < size; i++) {
7           sum += values[i];
8       }
9       return sum;
10  }
11
12  int main()
13  {
14      int values[] = {1, 2, 3, 4, 5};
15      int result = sum(values, sizeof(values) / sizeof(int));
16      printf("sum of values: %d \n", result);
17      return 0;
18  }
```

传参

- value copy
- 通过 指针 读取多个值

思考：line 3 和 line 14 的 values 是同一个变量?

返回值：1个，或者没有

思考：输出多个值怎样做?

Meaning?

Function: Basic Intro

```c
void find_max_n_min(int* values, int size, int* max, int *min)
{
    for (int i = 0; i < size; i++) {
        if (i == 0) {
            *max = values[i];
            *min = values[i];
        }
        else if (values[i] > *max) {
            *max = values[i];
        }
        else if (values[i] < *min) {
            *min = values[i];
        }
    }
}
```

输出多个值

- 方法1：传参，通过 指针输出多个结果
- 方法2：??

返回值：void （无）

关键点：

- 通过指针输出结果的时候，
- 先创建存放结果的变量
- 再调用函数，传入指向变量的指针

```c
int values[] = {1, 2, 3, 4, 5};
int size = sizeof(values) / sizeof(int);
int result = sum(values, size);
printf("sum of values: %d \n", result);
int max, min;
find_max_n_min(values, size, &max, &min);
printf("max: %d, min: %d\n", max, min);
```

Function: Basic Intro

```
28  int* generate_values(int size) {
29      int buf[100] = {0};
30      for (int i = 0; i < size; i++) {
31          buf[i] = (i + 1);
32      }
33      return buf;
34  }
35
36  int main()
37  {
38      int size = 5;
39      int* values = generate_values(size);
40      int result = sum(values, size);
41      printf("sum of values: %d \n", result);
```

输出多个值

- 方法1：传参，通过 指针 输出多个结果
- 方法2：**返回数组 (指针)？**

DON'T COPY THIS CODE!!

Function: Basic Intro

```
28  int* generate_values(int size) {
29      int* array = malloc(sizeof(int) * size);
30      for (int i = 0; i < size; i++) array[i] = (i+1);
31      return array;
32  }
33
34  int main()
35  {
36      int size = 5;
37      int* values = generate_values(size);
38      int result = sum(values, size);
39      printf("sum of values: %d \n", result);
40      free(values);
41      return 0;
42  }
```

输出多个值

- 方法1：传参，通过 指针输出多个结果
- **方法2：返回指针，动态分配内存**

注意释放内存！

Function: Basic Intro

```
28  void generate_values(int* array, int size) {
29      for (int i = 0; i < size; i++) array[i] = (i+1);
30  }
31
32  int main()
33  {
34      int size = 5;
35      int values[100];
36      generate_values(values, size);
37      int result = sum(values, size);
38      printf("sum of values: %d \n", result);
39      return 0;
40  }
```

输出多个值

- 方法1：传参，通过 指针输出多个结果
- 方法2：返回指针，动态分配内存

Which better ?

More：function design

SUSTech Southern University of Science and Technology

```
7   /**
8    * @brief Extract integers from a string
9    *
10   * Extract integers from the given string based on the specified delimiter,
11   * and store them in an integer array.
12   *
13   * @param str The input string
14   * @param delimiter The delimiter character
15   * @param count A pointer to store the count of extracted integers (output parameter)
16   *
17   * @return A pointer to the array containing the extracted integers
18   *         If no integers can be extracted or an error occurs, NULL is returned.
19   *
20   * @note The caller is responsible for managing the memory of the returned integer array
21   *       and ensuring it is freed after use.
22   */
23  int* extract_integers_from_string(const char* str, const char delimiter, int* count)…
```

More：function design

SUSTech Southern University of Science and Technology

```
8   /**
9    * @brief Case-insensitive keyword search
10   *
11   * Searches for the specified keyword (case-insensitive) in the given string and returns
12   * the indices of the top_n matching positions.
13   *
14   * @param str The string to search in
15   * @param keyword The keyword to search for
16   * @param top_n The maximum number of matching positions to return
17   * @param positions An integer array to store the indices of matching positions (output parameter)
18   *
19   * @return The number of matching keywords found
20   *         If no matches are found or an error occurs, a zero is returned.
21   *
22   * @note The size of the positions array should be at least top_n to ensure all returned indices can be stored.
23   *       If fewer than top_n matches are found, the unused positions in the positions array will remain undefined.
24   *       The caller should ensure that no further access to the elements in the positions array is needed after use.
25   */
26  int search_keyword_case_insensitive(const char* str, const char* keyword, int top_n, int* positions)
```

Outline

SUSTech Southern University of Science and Technology

- Review
- Function: Basic Intro
- **Assignment**

Assignment 1) 简易计算器

SUSTech Southern University of Science and Technology

Write a program: simple calculator, capable of handling arithmetic expressions

- input: a string, as arithmetic expressions (eg, 1+1)
  - including 2 kinds of arithmetic expressions: unary and binary operators
  - binary operators (二元运算): a+b, a-b, a*b, a/b, a^b
  - unary operators (一元运算): cos(a), sin(a), log(a), ln(a)
  - More: a^b, is a to the power of b. for cos/sin, a in degree. log (or, ln) is the logarithm base 10 (or, e)
- output: result of arithmetic expressions, print it out with 5 decimal places. (eg, 2.00000)
  - when expressions has error, or operator is unknown, print out “error”

- **Hints**: code template is given. Need to implement function that missing, or improve current function design.

Input

```
100+100
200.00000
```

```
-100-100
-200.00000
```

```
-100*1.23
-123.00000
```

```
-100/(-10)
10.00000
```

```
10^6
1000000.00000
```

Input

```
sin(30)
0.50000
```

```
cos(360)
1.00000
```

```
log(100)
2.00000
```

```
ln(2.71828)
1.00000
```

```
ln(-1)
error
```

Assignment

define operator tags

```
#define OP_NULL 0
#define OP_ADD  1
#define OP_SUB  2
#define OP_MUL  3
#define OP_DIV  4
// exponential operation
#define OP_EXP  5
#define OP_SIN  6
#define OP_COS  7
#define OP_LOG  8
#define OP_LN   9
```

```
18  char* get_input_str();
19
20  int get_operator(char* in_str);
21
22  void get_operand(char* in_str, int op, double* a, double* b);
23
24  int main()
25  {
26      char* in_str = get_input_str();
27
28      int op = get_operator(in_str);
29
30      double a, b;
31      get_operand(in_str, op, &a, &b);
32
33      // NEXT TODO
34      //      1) design function to perform operation
35      //      2) print out result
36
37      return 0;
38  }
```

// Note - 仅供参考，还需调整优化!

Highlight

- Readability
- Modulization

Appendix, some functions in <math.h>

SUSTech Southern University of Science and Technology

https://en.cppreference.com/w/c/numeric/math

Learn to search for standard library functions and use them correctly

Exponential functions

exp **expf** (C99) **expl** (C99)	computes *e* raised to the given power (e^x) (function)
exp2 (C99) **exp2f** (C99) **exp2l** (C99)	computes *2* raised to the given power (2^x) (function)
expm1 (C99) **expm1f** (C99) **expm1l** (C99)	computes *e* raised to the given power, minus one ($e^x - 1$) (function)
log **logf** (C99) **logl** (C99)	computes natural (base-*e*) logarithm ($\ln x$) (function)
log10 **log10f** (C99) **log10l** (C99)	computes common (base-*10*) logarithm ($\log_{10} x$) (function)
log2 (C99) **log2f** (C99) **log2l** (C99)	computes base-2 logarithm ($\log_2 x$) (function)
log1p (C99) **log1pf** (C99) **log1pl** (C99)	computes natural (base-*e*) logarithm of 1 plus the given number ($\ln(1+x)$) (function)

Trigonometric functions

sin **sinf** (C99) **sinl** (C99)	computes sine ($\sin x$) (function)
cos **cosf** (C99) **cosl** (C99)	computes cosine ($\cos x$) (function)
tan **tanf** (C99) **tanl** (C99)	computes tangent ($\tan x$) (function)
asin **asinf** (C99) **asinl** (C99)	computes arc sine ($\arcsin x$) (function)

Highlight: Reusability

Assignment 2)

Make Assignment #1 better ☺

五一假期快乐!

SUSTech Southern University of Science and Technology

Consider of special cases:

- When input invalid, ….
- When arithmetic overflow, …

```
1+
error
```

```
10^1000
error
```

Make your code readable to others

OJ 初步测试

+

现场 Code Review

SUSTech Southern University of Science and Technology

THANK YOU