N찍기

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include<stdio.h>  int main()  {  int i, j, st,a ;  scanf("%d", &st);    for (i = 0; i < st; i++) {  printf("\*");  if (i > 1) {  for (a = 0; a < i - 1; a++) {  printf(" ");  }  }  if (i > 0 && i < st) {  printf("\*");  }  for (j = 1; j < (st - i) - 1; j++) {  printf(" ");  }  if (i+1==st)  continue;  printf("\*\n");  }  } |



Wave

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  #include <string.h>  int main() {  char arr[100];  int i, length;  scanf("%[^\n]s", arr);  length = strlen(arr);  for (i = 0; i < length; i++) {  if (arr[i] == ' ') {  continue;  }  arr[i] -= 32;  printf("%s ", arr);  arr[i] += 32;  }  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

팩토리얼!

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include<stdio.h>  int num = 1;  void number(int a) {  num \*= a;  if (a == 0) {  printf("1");  return;  }  if (a == 1) {  printf("%d", num);  return;  }  number(a - 1);  }  int main() {  int a;  printf("INPUT NUMBER: ");  scanf("%d", &a);  number(a);  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

2진수

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include<stdio.h>  int arr[100];  void number(int count, int num) {  int a = num % 2;  arr[count]=a;  count--;  if (num <= 1) {  for (int i = count; i<=99 ; i++) {  if (arr[i] == 0 || arr[i] == 1)  printf("%d", arr[i]);  }  return;  }  number(count, num / 2);  }  int main() {  for (int j = 0; j < 100; j++) {  arr[j] = 5;  }  int count = 99;  int num = 0;  scanf("%d", &num);  number(count, num);  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

피보나치 수열

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  int sum(int num){  if (num == 0) {  return 0;  }  if (num == 1) {  return 1;  }  else  return sum(num-2) + sum(num-1);  }  int main(){  int num= 0;  scanf("%d", &num);  printf("%d ", sum(num));    } |

텍스트이(가) 표시된 사진

자동 생성된 설명

**#1. memset, strlen, strcmp, strcat, strcpy, strrev, strtok, strstr, atoi, itoa 함수 조사 및 실습**

**1) memset:** 메모리의 내용(값)을 원하는 크기만큼 특정 값으로 세팅할 수 있는 함수

**함수 원형 : void\* memset(void\* ptr, int value, size\_t num);**

Void\*ptr – 세팅하고자 하는 메모리의 시작 주소

Int value – 메모리에 세팅하고자 하는 값

Size\_t num – 길이 (바이트 단위로써 메모리의 크기 한조각 단위의 길이)

헤더 파일 : memory.h / string.h

+) memset 함수에 설정할 크기를 지정할 때 보통 숫자 대신에 sizeof를 사용함

|  |
| --- |
| #include<string.h>  #include<stdio.h>  int main(void)  {  char arr1[] = "catcertjjang";  memset(arr1, 'z', 3 \* sizeof(char));  printf(arr1);  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

Arr1 배열의 앞 세글자를 ‘z’라는 특정값으로 세팅함

**2) strlen:** const char\* 타입의 문자열을 받아서 해당 문자열의 길이를 반환하는 함수

**함수원형 : size\_t strlen(const char\* str);**  
size\_t – 객체나 값이 포함할 수 있는 최대 크기의 데이터를 표현하는 데이터 타입

헤더파일 - C언어 : **<string.h>** / C++ : **<cstring>**

Char\* 가 가리키는 주소에서부터 시작해서 ‘\0’이 나올 때까지의 문자의 개수를 센 후에 최종 길이를 반환

+) \0 : 문자열의 끝을 알려줌

|  |
| --- |
| #include <stdio.h>  #include <string.h>  int main()  {  char s1[10] = "cat";  char s2[10] = "cert";    printf("s1 문자열 길이 : %d\n", strlen(s1));  printf("s2 문자열 길이 : %d\n", strlen(s2));  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

|  |
| --- |
| #include <stdio.h>  #include <string.h>  int main()  {  char arr[20] = "cccat\0cert";  printf("arr 문자열 길이 : %d\n", strlen(arr));  return 0;  } |

텍스트이(가) 표시된 사진

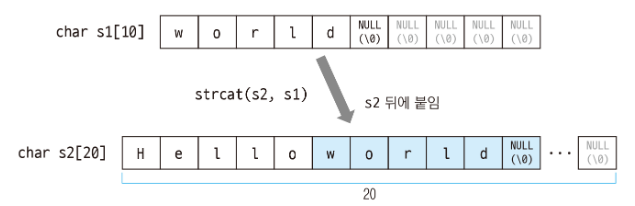
자동 생성된 설명

중간에 \0이 있기에 문자열의 끝으로 판단하고 그 전의 문자 개수만 셈

**3) strcat:** 문자열A에 원하는 문자열B를 문자열A 뒤쪽에 이어 붙이는 함수

**함수 원형 :** **char\* strcat(char\* dest, const char\* origin);**

**dest :** 복사를 받을 대상의 시작 주소  
**src :** 복사를 할 원본의 시작 주소  
**num :** 복사를 할 문자의 개수  
  
헤더파일: <string.h>    //C++ : <cstring>



+) 문자열 포인터로 문자열을 붙일 때는 함수로 동적 메모리를 할당 후, 공간 넉넉하게 설정

malloc함수 : void\* malloc(size\_t size)

size\_t 로 동적으로 할당할 메모리의 크기를 정해줌

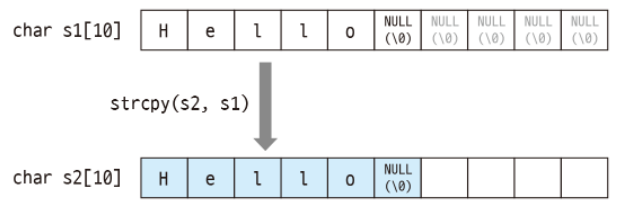
|  |
| --- |
| #include <stdio.h>  int main()  {    char str1[10] = "abc";  char str2[] = "def";  char str3[] = "ghi";    strcat(str1, str2);  printf("strcat : %s\n", str1);  return 0;  } |

**4) strcpy:** 대상 문자열 전체를 복사함

+ strncpy : 대상 문자열을 일정 길이만큼 복사한다

**함수원형 : char\* strcpy(char\* dest, const char\* src);**

dest : 복사를 받을 대상의 시작 주소  
src : 복사를 할 원본의 시작 주소  
num : 복사를 할 문자의 개수



복사된 결과가 배열에 저장될 때 배열의 크기는 NULL까지 들어갈 수 있어야함

→ 배열의 크기는 입력하고 싶은 문자열보다 최소 +1 로 설정

읽기 전용 메모리에는 문자열을 복사할 수 없음

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  #include <string.h>  int main()  {  char arr[5] = "cat";  char arr2[5];  strcpy(arr2, arr);  printf("%s\n", arr2);  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

**5) strrev:** 문자열을 거꾸로 뒤집는 함수

매개변수로 받은 문자열을 그대로 뒤집어서 반환하기 때문에 수정할 수 있는 형태의 문자열이여야함.

(문자열리터럴 수정 X)

**함수원형 :** char\*strrev(char\*string);

Char\*string: 거꾸로 뒤집혀질 문자열

테이블이(가) 표시된 사진

자동 생성된 설명

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  int main()  {  printf("%s", strrev("cat cert jjang"));  } |

매개변수로 받는 것은 수정할 수 있는 형태의 문자열이여야함

→ 수정가능한 문자 배열을 넣어줘야함

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  int main()  {  char a[30] = "Cat cert jjang";  printf("%s\n", strrev(a));  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

6) strtok: 문자열을 토큰처럼 조각내는 함수

**함수원형 : char\* strtok(char\* str, char\* delimiters);’**

Str: 자르고자 하는 문자열

Delimiters: 자를 기준을 정하는 구분자

Char\* 타입의 문자열 str을 첫번째 매개변수로 받음

→ 두번째 매개변수로 들어온 char\* 타입의 구분자를 기준으로 문자열을 잘라 문자열의 포인터를 하나씩 반환

Strtok 함수는 잘린 문자열을 한번에 얻을 수 없기에 while 반복문으로 문자열을 계속 자르다가 문자열이 나오지 않으면 반복문을 끝내는 방식으로 사용

→ 더 이상 자를 수 없으면 null 값을 리턴함으로 null 값이 리턴 될 때까지 반복해서 문자열을 자름

|  |
| --- |
| #include<stdio.h>  int main(void)  {  char str[] = "Cat,cert,Kim,min,yeong";  char\* ptr = strtok(str, ",");  while (ptr != NULL)  {  printf("%s\n", ptr);  ptr = strtok(NULL, ",");  }  return 0;  } |

+ 콤마를 기준으로 문자열을 자름

텍스트이(가) 표시된 사진

자동 생성된 설명

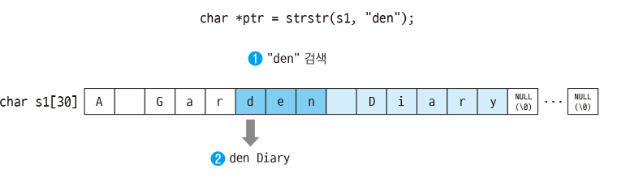
**7) strstr**: 특정 문자열에 원하는 문자열이 포함되어 있는지 확인하는 함수

**함수원형 :** char \*strstr(char \* const \_String, char const \* const \_SubString);

Str1: search 대상 문자열

Str2: 찾고자 하는 문자열

리턴값: 해당문자열로 시작하는 문자열의 위치 포인터



Strstr 함수는 대소문자를 구분함

While 반복문을 사용해서 문자열을 계속 검색할 수도 있음

|  |
| --- |
| #include <stdio.h>  #include <string.h>  int main()  {  char s1[30] = "cat cert";  char\* ptr = strstr(s1, "cat");  printf("%s\n", ptr);  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

**8) atoi** = char to int = 문자열을 정수 타입으로

**함수원형 :** int atoi (const char\* cStr);

매개변수로 들어온 문자열을 앞에서부터 읽어서 “공백” or “숫자가 아닌 문자”가 올 때까지 숫자로 변환해주는 원리

Atoi 함수는 소수점이나 지수를 인식하지 못함

입력 문자를 숫자로 해석해서 생성되는 int값을 리턴함 → 함수가 입력을 변환할 수 없는 경우 리턴값은 0

“0”을 atoi 한 결과와 구분되지 않기에 주의해야함

|  |
| --- |
| #include <stdio.h>  int main()  {  char s1[20] = "20220523";  int num1;  num1 = atoi(s1);  printf("%d\n", num1);  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

정수형태로 출력했는데 정상 출력됨을 확인

**9) itoa:** 정수형을 문자열로 (비 표준함수)

integer 값을 2진, 8진, 10진, 16진 문자열

**함수원형 : char** \* **itoa**(**int** val, **char** \* buf, **int** radix)

변환할 정수 / 변환받을 문자열을 저장할 문자열 변수 / 변환할 진수 (10: 10진수)

Visual studio 에서만 정의 해놓은 함수 ( 다른 플랫폼에서 사용X)

|  |
| --- |
| #include <stdio.h>  int main()  {  int a = 523;  char str[10] = "2020";  char tmp[10] = { 0, };  strcat(str, itoa(a, tmp, 10));  printf("%s\n", str);  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

**10) strcmp :** **두개의 문자열을 비교 / string1과 string2 두 문자열을 비교함**

**원형 함수 : int strcmp(const char\* str1, const char\* str2)**

첫번째 매개변수 str1 : 비교할 문자열1

두번째 매개변수 str2 : 비교할 문자열2

매개 변수로 들어온 두개의 문자열을 비교해서 문자열이 완전히 같으면 0을 반환 / 다르면 음수 혹은 양수를 반환

대소문자 구분 O / 각 문자별 숫자가 정해져있으므로 대소 비교 가능

**(1) str1 < str2 인 경우에는 음수 반환**

**(2) str1 > str2 인 경우에는 양수 반환**

**(3) str1 == str2 인 경우에는 0을 반환**

|  |
| --- |
| #include <stdio.h>  #include <string.h>  int main()  {  char s1[10] = "cat";  char\* s2 = "cat";  int ret = strcmp(s1, s2);  printf("%d\n", ret);  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

“cat”=”cat” 이므로 0 출력

**#2 위 문자열 함수 중 리눅스 환경에서 실행되지 않는 함수 조사**

**Strrev 함수 (우분투에도 없음)**

**Itoa 함수 (비 표준함수)**