20191017 실습 5주차

박 건 호

KITRI BoB 8th

5422 RTDCS

devgunho.github.io

1. 문자열에서 가장 큰 곱 찾기 (10)

- 1번째 입력값 3 : 찾고싶은 연속되는 숫자의 길이
- 2번째 입력값 123456789 : 숫자 문자열(string)
- 이후 출력값 :

찾고싶은 연속되는 숫자의 길이는 3이고,
이후 그 길이만큼의 숫자의 곱을 출력한 예시이다.
첫 연속 3자리의 곱은 123 => 1*2*3 => 6
234 => 2*3*4 => 24
...
789 => 7*8*9 => 504

즉 입력 3과 123456789에 대해서 504를 출력하면 올바른 답이다.

```
3
123456789
123
6
234
24
345
60
456
120
567
210
678
336
789
504
```

HINT : str.c_str(); // string을 char*로 변환 이후 char->int로 Type casting
이유 : string에서 int로 바로 Type casting 불가

```
Microsoft Visual Studio Debug Console
3
111111111111111111111111111111111972222222
126

D:₩_Workspace₩19_ComputerProgramming₩CP_Lectur
Press any key to close this window . . .
```

2. SquCirTriDiff 클래스 디자인하기 (10)

SquCirDiff 클래스

- private 멤버변수

  double 길이 // 원의 경우 **'반지름'**, 정사각형의 경우 **'한 변의 길이/2'**

  // 삼각형의 경우 밑변과 높이가 정사각형의 한변의 길이

- public 멤버함수

  SetLength(double length) - 값 대입

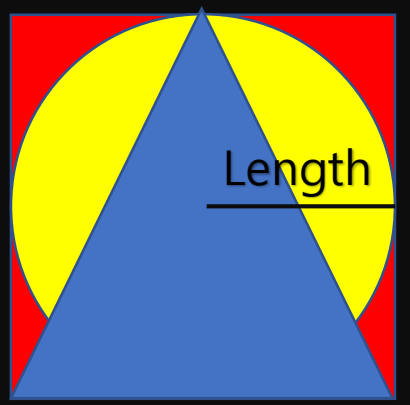
  GetSquRound() – 정사각형의 둘레 리턴

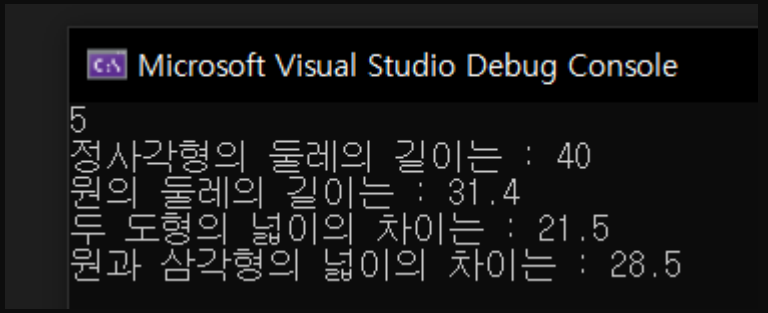
  GetCirRound() – 원의 둘레 리턴

  GetSquCirAreaDiff() – 정사각형에서 원의 넓이 차이 리턴

  // PI 값은 3.14로 고정

  GetCirTriAreaDiff() – 원에서 삼각형의 넓이 차이 리턴





3. 단어 수 세기 (10)

```
Microsoft Visual Studio Debug Console
my name is joker
4
```

class CountString 설계 요구사항

```
Microsoft Visual Studio Debug Console
     asdf      asfd    asdf  asdf as asdf
6

D:₩_Workspace₩19_ComputerProgramming₩CP_Lecture
Press any key to close this window . . .
```

private :

char* s;

int size:

public :

CountString(string str) // str의 길이 만큼 new(동적할당)하고 strcpy 객체 복사

~CountString() // 생성자에서 동적할당한 메모리 해제

getWordCnt() // 입력받은 문자열에서 단어의 개수 출력

3. 단어 수 세기

5장 12page 참고

setter 없이 생성자에서
private 멤버변수로 strcpy

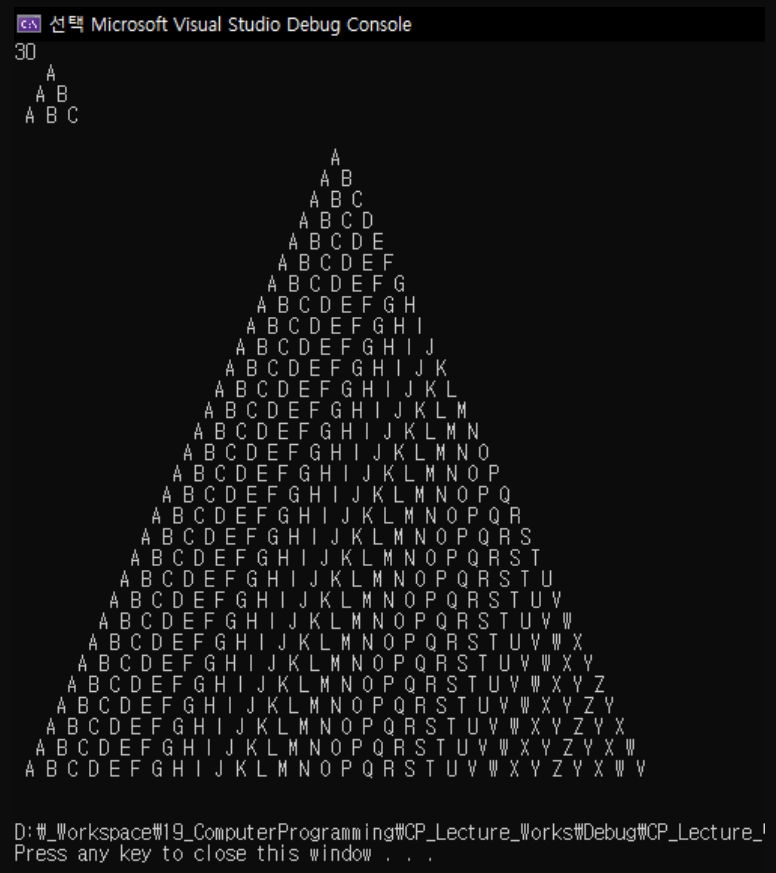
```
#include <string.h>

class MyString {
private:
        char *s;
        int size;
public:
        MyString(char *c) {
                size = strlen(c)+1;
                s = new char[size];
                strcpy(s, c);
        }
        ~MyString() {
                delete[] s;
        }
};
int main() {
        MyString str("abcdefghijk");
}
```

4. 알파벳 삼각형 출력 클래스 (10)

- 기본적으로 객체를 생성하면 3줄짜리 알파벳 삼각형을 출력한다.
- 사용자로부터 출력하고 싶은 줄 수를 입력받으면 입력받은 수만큼의 높이의 삼각형을 출력한다.

```
class alphaTri
private:
        int heigth;
public:
        alphaTri()          // 기본 생성자
        void setHeight(int val)
        void printAlphaTri()
```



Test EX1.

```
int do(int a[], int n) {
      int j, s = 0;
      for(j = 0; j < n; j++)
            s += *(a+j);
      return s/n;
}
int main(void) {
      int a[10] = {1, 2, 3, 4, 5, 6, 7,};
      printf("%d\n", do(a, 10));    _____
      printf("%d\n", do(a+3, 5));   _____
      printf("%d\n", do(&a[2], 4)); _____
}
```

Test EX2.

주소	메모리 내용	
_____	1.0	B[0]
_____	2.0	B[1]
1360	3.0	B[2]
_____	4.0	B[3]
	.....	
2000		B
2004		p

```
double B[5] = {1.0, 2.0, 3.0, 4.0};
double *p = B+2, *q = B;
printf("%d\n", B); _____
printf("%lf\n", B[2]-*p); _____
printf("%lf\n",*(p+1)**(p-1)); _____
p++;
printf("%d\n", &B[3]); _____
printf("%d\n", p - q); _____
```

Test EX3.

```
char S1[100] = “Truth”, S2[] = “over there”;
printf(“%d\n”, strlen(S1)+strlen(S2));  _____________
printf(“%s\n”, S1+2); _____________
printf(“%s\n”, strcat(S1, strcat(“ is ”, S2+5)) );

_____________
printf(“%s\n”, “C-Language”+2); _____________
printf(“%d\n”, strcmp(S2, S1)); _____________
```

struct (Allocating struct)

```c
#include <stdlib.h>

struct A
{
    int one;
    int two;
};

int main()
{
    // dynamically allocated struct
    struct A *pA = (struct A *)malloc(sizeof(struct A));
    (*pA).one = 1;
    pA -> two = 2;

    // statically allocated struct
    struct A a;
    a.one = 1;
    a.two = 2;

    return 0;
}
```

struct (Using typedef)

```c
#include <stdlib.h>

struct A
{
    int one;
    int two;
};

typedef struct A A_t;

int main()
{
    A_t *pA = (A_t *)malloc(sizeof(A_t));
    (*pA).one = 1;
    pA -> two = 2;

    A_t a;
    a.one = 1;
    a.two = 2;
}
```

이중 포인터를 활용한 2차원 배열

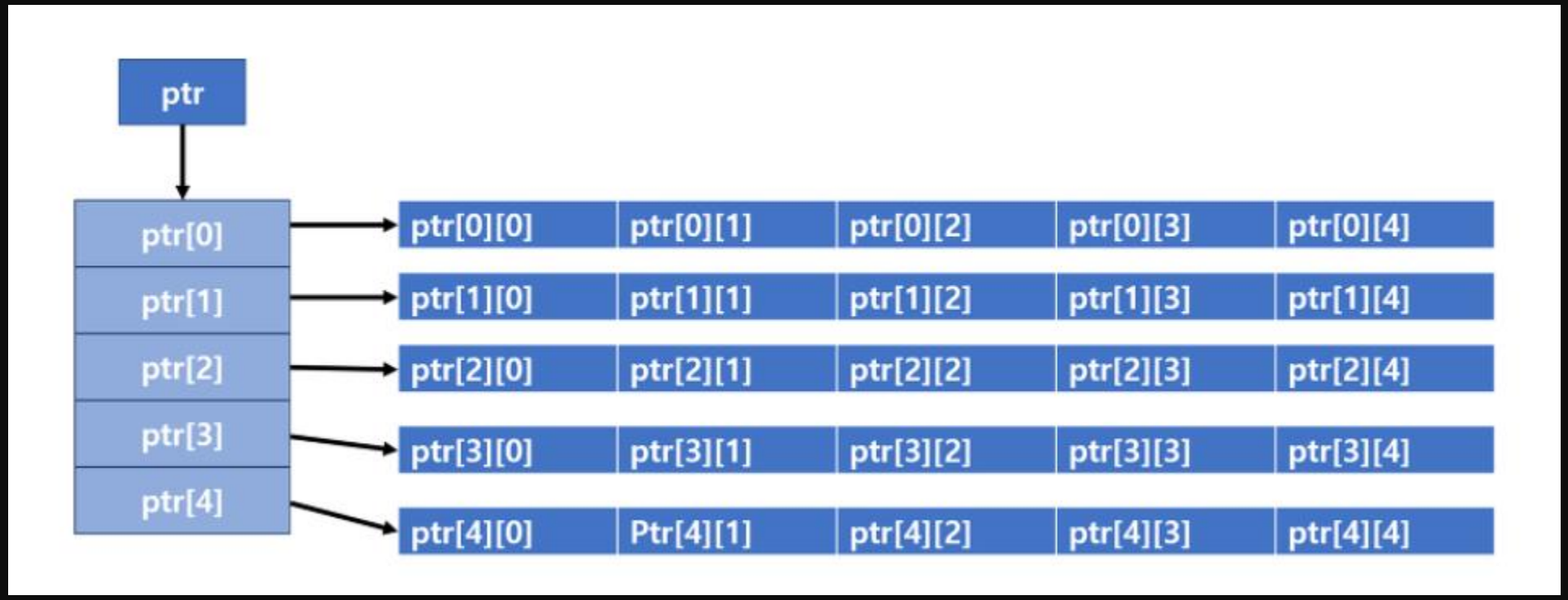
C

```c
int* ptr = NULL;
ptr = (int*)malloc(sizeof(int)*5);
```

C++

```cpp
int* ptr = nullptr;
ptr = new int[5];
```

이중 포인터를 활용한 2차원 배열



이중 포인터를 활용한 2차원 배열

C

```c
int** ptr=NULL;
ptr = (int**)malloc(sizeof(int*)*5);
for(int i=0;i<5;i++)
   ptr[i] = (int*)malloc(sizeof(int)*5);
```

C++

```cpp
int** ptr = nullptr;
ptr=new int*[5];
for(int i=0;i<5;i++)
   ptr[i]=new int[5];
```

이중 포인터를 활용한 2차원 배열

- 할당 후 해제하기
- 생성할 때와 역순으로 해제

C

```c
for (int i = 0; i < 5; i++)
        free(ptr[i]);
free(ptr);
```

C++

```cpp
for (int i = 0; i < 5; i++)
        delete ptr[i];
delete ptr;
```