1번 문제 제출완료

Revise the given source code properly so that it swaps values of *na* and *nb*.

입/출력 예시

␣ : 공백 ↵ : 줄바꿈 ⇥ : 탭

예시 1

출력

```
Before␣Swap␣:␣na␣=␣1,␣nb␣=␣99↵
↵
After␣␣Swap␣:␣na␣=␣99,␣nb␣=␣1↵
↵
```

* 입출력 형식을 잘 지켜주세요

2번 문제 제출완료

Complete mystrlen(), mystrcpy(), and mystrcmp() functions in the given source code.

You are not allowed to use library functions.

int mystrlen(char *str) : Returns the length of the given null-terminated byte string, that is, the number of characters in a character array whose first element is pointed to by str up to and not including the first null character.

- Parameters
  - str - pointer to the null-terminated byte string to be examined
- Return value
  - The length of the null-terminated byte string str

char * mystrcpy(char *dest, char *src) : Copies the null-terminated byte string pointed to by src, including the null terminator, to the character array whose first element is pointed to by dest.

- Parameters
  - dest - pointer to the character array to write to
  - src - pointer to the null-terminated byte string to copy from
- Return value
  - returns a copy of dest

int mystrcmp(char *lhs, char *rhs) : Compares two null-terminated byte strings lexicographically. The sign of the result is the sign of the difference between the values of the first pair of characters (both interpreted as unsigned char) that differ in the strings being compared.

- Parameters
  - lhs, rhs - pointers to the null-terminated byte strings to compare
- Return value
  - A negative value if lhs appears before rhs in lexicographical order.
  - Zero if lhs and rhs are equal.
  - A positive value if lhs appears after rhs in lexicographical order.

3번 문제 제출완료

Complete mystrrev() in the given source code.

While you are not allowed to use library functions, you can add and use your own functions.

char *mystrrev(char* str) : reverse the null-terminated byte string pointed to by str;

- Parameters
  - str - pointer to the null-terminated byte string to reverse
- Return value
  - returns a copy of str.

입/출력 예시

␣ : 공백 ↵ : 줄바꿈 →⇤ : 탭

예시 1

입력

```
1234567890
```

출력

```
1234567890␣␣␣␣␣␣␣␣␣␣:␣After␣Reverse␣>>>↵
0987654321␣␣␣␣␣␣␣␣␣␣↵
```

* 입출력 형식을 잘 지켜주세요

4번 문제 제출완료

Complete mystrstr() in the given source code.

While you are not allowed to use library functions, you can add and use your own functions.

char *mystrstr(char* str, char* substr) : Finds the first occurrence of the null-terminated byte string pointed to by substr in the null-terminated byte string pointed to by str. The terminating null characters are not compared.

- Parameters
  - str - pointer to the null-terminated byte string to examine
  - substr - pointer to the null-terminated byte string to search for
- Return value
  - Pointer to the first character of the found substring in str, or **a null pointer if a such substring is not found**. If substr points to an empty string, str is returned.

입/출력 예시

␣ : 공백 ↵ : 줄바꿈 ⇥ : 탭

예시 1

출력

```
found␣the␣string␣'two'␣in␣'one␣two␣three'␣at␣position:␣4↵
found␣the␣string␣''␣in␣'one␣two␣three'␣at␣position:␣0↵
the␣string␣'nine'␣was␣not␣found␣in␣'one␣two␣three'↵
found␣the␣string␣'n'␣in␣'one␣two␣three'␣at␣position:␣1↵
```

* 입출력 형식을 잘 지켜주세요

5번 문제 제출완료

Palindrome

A word, phrase, or sequence that reads the same backward as forward, e.g., madam or racecar.

Sentence-length palindromes may be written when allowances are made for adjustments to capital letters, punctuation, and word dividers, such as "A man, a plan, a canal, Panama!", "Was it a car or a cat I saw?" or "No 'x' in Nixon".

int ispalindrome(char *str) : Function to determine if a string is a palindrome.

- Parameter
  - str: a pointer to a string to check for palindrome
- Return value
  - 1 if palindrome, 0 otherwise.

You can use standard string and character processing functions such as *strlen()* declared in <string.h> and *isalpha()* declared in <ctype.h>

입/출력 예시

␣ : 공백 ↵ : 줄바꿈 →| : 탭

예시 1

입력

```
1234␣␣5␣4321
```

출력

```
[1234␣␣5␣4321]␣is␣a␣palindrome↵
```

예시 2

입력

```
A␣man,␣a␣plan,␣a␣canal,␣Panama!
```

출력

```
[A␣man,␣a␣plan,␣a␣canal,␣Panama!]␣is␣a␣palindrome↵
```

예시 3

입력

```
1␣2␣3␣0␣0␣2␣1
```

출력

```
[1␣2␣3␣0␣0␣2␣1]␣is␣not␣a␣palindrome↵
```

* 입출력 형식을 잘 지켜주세요

6번 문제 제출완료

Revise the given source code properly so that we can get the length, the maximum, and the minimum of an integer array pointed by *pnums* using the "get_len_max_min()" function.

You can safely assume that all item values in *nums* are positive except the last one to indicate the end of the array. Also, there is at least one valid item in *nums*.

The length of an array is the number of valid items in the array excluding the last one with "END_MARK". For example, the length of [1, 2, 3, END_MARK] is 3.

You are not allowed to add new variables or functions to the code.

7번 문제 제출완료

[Game of Stone 2]

Two players called Alice and Bob are playing a game with a number of stones. Alice always plays first, and the two players move in alternating turns. The game's rules are as follows:

- In a single move, a player can remove either 1, 2, or 3 stones from the game board.
  - But there is **a restriction**. If the number of stones removed from the previous move of the opponent is ***p***, the player **can not move *p*** stones.
  - The first move of Alice is free from the above restriction because there is no previous move of Bob.
- If a player is unable to make a move because there are not enough stones, that player loses the game.

For example, after removing 2 stones by Bob, now there remain 4 stones. They can make the following moves:

- Alice removes 3 stones leaving 1 stone. Bob removes 1 stone and wins.
  - Alice cannot remove 2 stones because of restrictions.
- Alice removes 1 stone and leaves 3 stones. Bob removes 3 stones and wins.

So Alice cannot win whatever number she chooses.

Complete 2 functions, win() and Alice_Move2(), in the given source code. Your code should work fast enough to solve the problem within the timeout.

int win(int n, int p) : **determines whether a player can win** when the number of current stones is n and the number of stones removed in the previous opponent's turn is p.

- Parameter
  - n - the number of stones remaining
  - p - the number of stones removed in the previous opponent's turn
- Return value
  - For n > 0, return 1 if a player can win, 0 otherwise
  - For n==0, return 0
  - For n < 0, return 1

- For example, win(4,2) should return 0 (Lose) as explained above.
- static int b_win[4][MAX_STONES+1] : a 2-dimensional static integer array that holds determined results. the first index stands for p, and the second index stands for n. That is, b_win[2][4] should have a value when p = 2 and n = 4. Of course, it should hold 0 as explained previously.

(win() 함수는 현재 남은 돌이 n, 상대가 바로 전에 옮긴 돌의 수가 p인 상황에서 승리할 수 있는지 여부를 판단하여 승리할 수 있는 경우 1을 그렇지 못한 경우 0을 return하여야 한다. 이 함수를 작성하기 위해서 당신은 win() 함수 내의 2차원 static int 배열인 b_win[4][MAX_STONES+1]의 값을 적절하게 채워 나가야 한다. 이 배열의 첫 index는 p 값을, 두번째 index는 n 값을 의미한다. 즉 b_win[2][4]는 p=2, n=4인 경우의 승부 결과 값을 가져야 하며 앞의 예 설명에서 알 수 있듯이 이는 0이다.)

int Alice_Move2(int n_x) : returns the largest number of stones to remove in Alice's turn if she can win. Otherwise, that is, **if she can't win, return the smallest number of stones that she can remove.** The number of current stones remaining is maintained in the global variable ***gn_stones*** which is accessible inside Alice_Move2(). And the number of stones removed from Bob's previous move is given through the parameter ***"n_x"***.

You can safely assume that Alice_Move2() is called only when she can remove stones.

(Alice가 제거할 돌의 수를 결정하는 함수 Alice_Move2()를 작성하라. 이 함수는 Alice가 이길 수 있는 상황에서 이기는 상황을 계속 유지시키는 최대 값을 반환한다. Alice가 이길 수 없는 상황의 경우 제거할 수 있는 최소 값을 반환하여야 한다. 참고로 Alice_Move2()는 항상 돌을 제거할 있는 상황에서 호출된다고 가정하여도 된다)

입/출력 예시

␣ : 공백 ↵ : 줄바꿈 ⇥ : 탭

예시 1

입력

```
10
```

출력

```
The␣starting␣number␣of␣stones␣10↵
Alice>␣3␣removed␣:␣[10␣=>␣7␣stones␣remained]↵
Bob>>>␣2␣removed␣:␣[7␣=>␣5␣stones␣remained]↵
Alice>␣1␣removed␣:␣[5␣=>␣4␣stones␣remained]↵
Bob>>>␣2␣removed␣:␣[4␣=>␣2␣stones␣remained]↵
Alice>␣1␣removed␣:␣[2␣=>␣1␣stones␣remained]↵
Congratulations␣Alice,␣You␣Win␣!!!
```

예시 2

입력

```
65
```

출력

```
The␣starting␣number␣of␣stones␣65↵
Alice>␣1␣removed␣:␣[65␣=>␣64␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[64␣=>␣61␣stones␣remained]↵
Alice>␣1␣removed␣:␣[61␣=>␣60␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[60␣=>␣57␣stones␣remained]↵
Alice>␣1␣removed␣:␣[57␣=>␣56␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[56␣=>␣53␣stones␣remained]↵
Alice>␣1␣removed␣:␣[53␣=>␣52␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[52␣=>␣49␣stones␣remained]↵
Alice>␣1␣removed␣:␣[49␣=>␣48␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[48␣=>␣45␣stones␣remained]↵
Alice>␣1␣removed␣:␣[45␣=>␣44␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[44␣=>␣41␣stones␣remained]↵
Alice>␣1␣removed␣:␣[41␣=>␣40␣stones␣remained]↵
Bob>>>␣2␣removed␣:␣[40␣=>␣38␣stones␣remained]↵
Alice>␣3␣removed␣:␣[38␣=>␣35␣stones␣remained]↵
Bob>>>␣2␣removed␣:␣[35␣=>␣33␣stones␣remained]↵
Alice>␣1␣removed␣:␣[33␣=>␣32␣stones␣remained]↵
Bob>>>␣3␣removed␣:␣[32␣=>␣29␣stones␣remained]↵
Alice>␣1␣removed␣:␣[29␣=>␣28␣stones␣remained]↵
Bob>>>␣2␣removed␣:␣[28␣=>␣26␣stones␣remained]↵
Alice>␣3␣removed␣:␣[26␣=>␣23␣stones␣remained]↵
B
```

※ 입출력 형식을 잘 지켜주세요