Trạng thái	Đã xong
Bắt đầu vào lúc	Thứ Sáu, 18 tháng 4 2025, 11:22 AM
Kết thúc lúc	Thứ Sáu, 18 tháng 4 2025, 12:55 PM
Thời gian thực hiện	1 giờ 33 phút

Câu hỏi **1**

Đúng

Đạt điểm 1,00

[Tiếng Việt]

Một chuỗi được gọi là palindrome nếu chuỗi đó giống với chuỗi được đảo ngược từ chính nó. Ví dụ: “eye”, “noon”, “abcba”...

Hãy viết hàm kiểm tra xem một chuỗi có là palindrome hay không?

Đầu vào:

- const char* str: chuỗi cần kiểm tra palindrome. str chỉ bao gồm chữ cái thường

Đầu ra:

- bool: true nếu chuỗi str là palindrome, ngược lại false

[English]

A string is a palindrome if it reads the same forward and backward. For example: "eye", "noon", "abcba", ...

Write a function to check if a given string is a palindrome

Input:

- const char* str: the string to be checked. str only contains lowercase letters

Output:

- bool: true if str is a palindrome, false otherwise

For example:

Test	Result
`const char* str = "abba";` `cout << isPalindrome(str);`	1
`const char* str = "axz";` `cout << isPalindrome(str);`	0

Answer: (penalty regime: 0 %)

Reset answer

```
bool isPalindrome(const char* str) {
    // Write your code here
    int size = 0;
    while(str[size] != '\0')   size++;

    for(int i = 0; i < size / 2; ++i){
        if(str[i] != str[size - i -1])
            return false;
    }
    return true;
}
```

	Test	Expected	Got	
✓	`const char* str = "abba";` `cout << isPalindrome(str);`	1	1	✓
✓	`const char* str = "axz";` `cout << isPalindrome(str);`	0	0	✓

Passed all tests! ✓

Câu hỏi **2**

Đúng

Đạt điểm 1,00

[Tiếng Việt]

Một số tự nhiên n được gọi là đặc biệt khi và chỉ khi n là số nguyên tố và tổng các chữ số của n cũng là số nguyên tố. Viết hàm kiểm tra một số tự nhiên có đặc biệt hay không.

Đầu vào:

- int n: số tự nhiên cần kiểm tra có phải số đặc biệt không

Đầu ra:

- bool: trả về true nếu n là số đặc biệt, ngược lại trả về false

[English]

A natural number n is special if and only if n is a prime number and the sum of all the digits of n is also a prime number. Write a function that determines if a natural number is a special or not.

Input:

int n: a natural number n. 0 <= n <= 1000

Output:

bool: return true if n is special, return false otherwise

For example:

Test	Input	Result

`int n;`
`cin >> n;`
`cout << isSpecialNumber(n);`
	23	1

`int n;`
`cin >> n;`
`cout << isSpecialNumber(n);`
	7	1

Answer: (penalty regime: 0 %)

Reset answer

```
bool isPrime(int n){
    if(n < 2)   return false;

    for(int i = 2; i*i <= n; ++i){
     if(n % i == 0) return false;
    }
    return true;
}
bool isSpecialNumber(int n) {
    // Write your code
   if(!isPrime(n)) return false;

        int temp = n;
        int sum = 0;
        while(temp > 0){
            sum += temp % 10;
            temp /= 10;
        }

        return isPrime(sum);
}
```

	Test	Input	Expected	Got	
✓	int n; cin >> n; cout << isSpecialNumber(n);	23	1	1	✓
✓	int n; cin >> n; cout << isSpecialNumber(n);	7	1	1	✓

Passed all tests! ✓

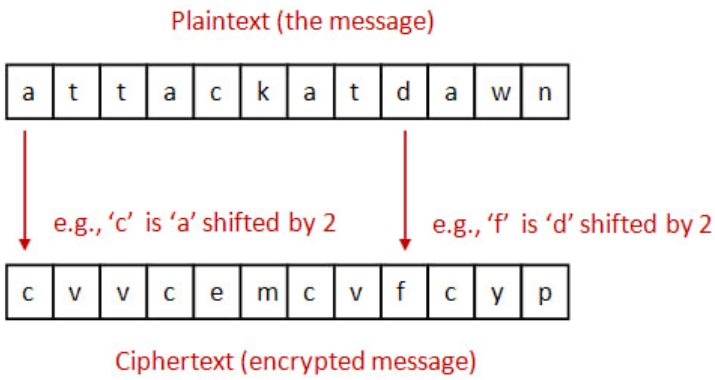
Câu hỏi **3**

Đúng

Đạt điểm 1,00

[Tiếng Việt]

Viết một hàm mã hóa và một hàm giải mã một đoạn text theo phương pháp Caesar Cipher. Để mã hoá và giải mã một chuỗi ký tự text, ta cần một tham số có giá trị nguyên là shift.



Hàm mã hóa (tên **encrypt**) sẽ thay đổi từng chữ cái trong text bằng cách dịch chuyển chữ cái đó sang phải shift lần trong bảng chữ cái. Ví dụ với shift = 3. Khi đó 'a' được mã hoá thành 'd', 'b' được mã hoá thành 'e',... 'z' được mã hoá thành 'c'.

Hàm giải mã (tên **decrypt**) sẽ nhận một chuỗi ký tự text và giá trị nguyên shift và giải mã chuỗi ký tự này thành chuỗi ban đầu (tức dịch chuyển từng chữ cái sang trái shift lần trong bảng chữ cái)

Đầu vào:

- **char* text**: chuỗi ký tự cần được mã hoá hoặc giải mã, chỉ bao gồm chữ cái thường và hoa
- **int shift**: giá trị dịch chuyển trong Caesar Cipher

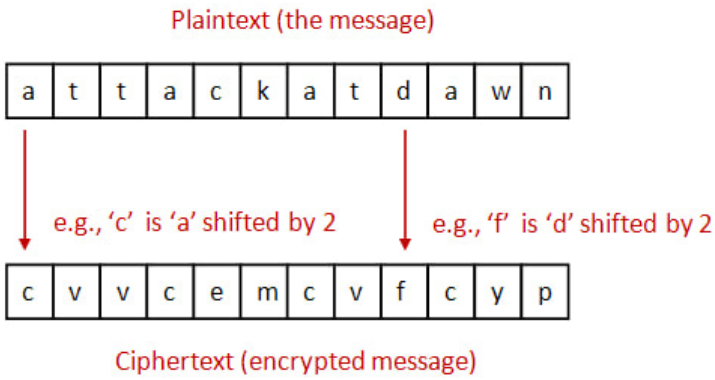
Đầu ra:

- Hàm không trả về.
- Chuỗi ký tự truyền vào **text** sẽ thay đổi trực tiếp trong hàm.

[English]

Write a function to encrypt and a function to decrypt a text string using Caesar Cipher technique.

In this technique, to encrypt a string we need a parameter of type integer called 'shift'.



The **encrypt** method will shift each letter by some fixed number of position (determined by the parameter 'shift') to the right in the alphabet. For example, when 'shift' is 3, 'a' will be replaced by 'd', 'b' will become 'e', ... , 'z' will become 'c'.

The **decrypt** method will receive an encoded string and a shift value and it will decode this string to get the original string, which means shifting each character to the left in the alphabet.

Input:

- **char* text:** the text string that needs to be encrypted or decrypted. text only contains lowercase and uppercase ASCII letters
- **int shift**: the shift value in Caesar Cipher technique

Output:

- The function returns nothing.
- The input parameter **text** will be updated in-place.

For example:

Test	Input	Result

`int n, shift;`
`cin >> n >> shift;`
`char* text = new char[n+1];`
`for(int i = 0; i < n; i++) cin >> text[i];`
`text[n] = 0;`
 	
`encrypt(text, shift);`
`cout << text << '\n';`
`decrypt(text, shift);`
`cout << text;`
 	
`delete[] text;`
	`6 3`
	`aczDYZ`
		`dfcGBC`
		`aczDYZ`

	`16 25` `programmingisfun`	`oqnfqzllhmfhretm` `programmingisfun`

```
int n, shift;
cin >> n >> shift;
char* text = new
char[n+1];
for(int i = 0; i < n;
i++) cin >> text[i];
text[n] = 0;

encrypt(text, shift);
cout << text << '\n';
decrypt(text, shift);
cout << text;

delete[] text;
```

Answer: (penalty regime: 0 %)

Reset answer

```
1  void encrypt(char* text, int shift) {
2      // Write your code
3      shift = (shift % 26 + 26) % 26;
4      char *p = text;
5      while(*p){
6          if(*p >= 'A' && *p <= 'Z')
7              *p = 'A' + (*p - 'A' + shift + 26) %
8          else if (*p >= 'a' && *p <= 'z')
9              *p = 'a' + (*p - 'a' + shift + 26) % 2
10
11             p++;
12     }
13 }
14
15 void decrypt(char* text, int shift) {
16     // Write your code
17     shift = (shift % 26 + 26) % 26;
18      char *p = text;
19     while(*p){
20         if(*p >= 'A' && *p <= 'Z')
21             *p = 'A' + (*p - 'A' - shift + 26) %
22         else if (*p >= 'a' && *p <= 'z')
23             *p = 'a' + (*p - 'a' - shift + 26) % 2
24
25             p++;
26     }
27 }
```

	Test	Input	Expected	C

✓	int n, shift;
	cin >> n >> shift;
	char* text = new char[n+1];
	for(int i = 0; i < n; i++) cin >> text[i];
	text[n] = 0;
	
	encrypt(text, shift);
	cout << text << '\n';
	decrypt(text, shift);
	cout << text;
	
	delete[] text;
		6 3
		aczDYZ
			dfcGBC
			aczDYZ
				d
				a

✓	int n, shift;
	cin >> n >> shift;
	char* text = new char[n+1];
	for(int i = 0; i < n; i++) cin >> text[i];
	text[n] = 0;
	
	encrypt(text, shift);
	cout << text << '\n';
	decrypt(text, shift);
	cout << text;
	
	delete[] text;
		16 25
		programmingisfun
			oqnfqzllhmfhretm
			programmingisfun
				o
				p

Passed all tests! ✓

Câu hỏi **4**

Đúng

Đạt điểm 1,00

[Tiếng Việt]

Viết hàm kiểm tra các phần tử trong mảng có duy nhất hay không

Đầu vào:

- int* arr: mảng số tự nhiên
- int n: số lượng phần tử trong mảng

Đầu ra:

- bool: trả về true nếu các phần tử trong mảng là duy nhất, ngược lại trả về false

Chú ý: arr[i] nằm trong khoảng từ [0, 1000]

[English]

Write a function that determines if the elements in the given array is unique

Input:

- int* arr: array of integer
- int n: the size of the array

Output:

- bool: return true if the elements in arr is unique, otherwise return false

Note: arr[i] is in the range of [0, 1000]

For example:

Test	Input	Result

`int n;`
`cin >> n;`
`int* arr = new int[n];`
`for(int i = 0; i < n; i++) {`
`  cin >> arr[i];`
`}`
`cout << checkElementsUniqueness(arr, n);`
`delete[] arr;`
	5
	2 5 13 5
	2
		0

`int n;`
`cin >> n;`
`int* arr = new int[n];`
`for(int i = 0; i < n; i++) {`
`  cin >> arr[i];`
`}`
`cout << checkElementsUniqueness(arr, n);`
`delete[] arr;`
	3
	17 10 25
		1

Answer: (penalty regime: 0 %)

Reset answer

```
bool checkElementsUniqueness(int* arr, int n) {
    // Write your code
    for(int i = 0; i < n; ++i){
        for(int j = i + 1; j < n; ++j)
        {
            if(arr[i] == arr[j])
                return false;
        }
    }
    return true;
}
```

	Test	Input	Expected	Got

✓	int n;
	cin >> n;
	int* arr = new int[n];
	for(int i = 0; i < n; i++) {
	    cin >> arr[i];
	}
	cout <<
	checkElementsUniqueness(arr,
	n);
	delete[] arr;
		5
		2 5
		13 5
		2
			0	0

✓	int n;
	cin >> n;
	int* arr = new int[n];
	for(int i = 0; i < n; i++) {
	    cin >> arr[i];
	}
	cout <<
	checkElementsUniqueness(arr,
	n);
	delete[] arr;
		3
		17
		10
		25
			1	1

Passed all tests! ✓

Câu hỏi **5**

Đúng

[Tiếng Việt]

Không chấm điểm

Cho một số thập phân dương làm đầu vào, chúng ta cần triển khai hàm

```
long int decimalToBinary(int decimal_number){}
```

để chuyển đổi số thập phân dương đã cho thành số nhị phân tương đương.

Xin lưu ý rằng bạn không thể sử dụng từ khóa for, while, goto (ngay cả trong tên biến, comment).

Đối với bài tập này, chúng ta có #include <iostream> và sử dụng namespace std;

[English]

Given a positive decimal number as input, we need to implement function

```
long int decimalToBinary(int decimal_number){}
```

to convert the given positive decimal number into equivalent binary number.

Please note that you can't using key work for, while, goto (even in variable names, comment).

For this exercise, we have #include <iostream> and using namespace std;

For example:

Test	Result
`cout << decimalToBinary(20);`	`10100`

Answer:

Reset answer

```
long int helper(int n, long int place){
    if(n == 0)  return 0;
    return (n % 2) * place + helper(n / 2, place *
}
long int decimalToBinary(int decimal_number)
{
    if(decimal_number == 0) return 0;
    return helper(decimal_number, 1);
}
```

	Test	Expected	Got	
✓	`cout << decimalToBinary(20);`	`10100`	`10100`	✓
✓	`cout << decimalToBinary(192);`	`11000000`	`11000000`	✓

Passed all tests! ✓