**CAESAR**

Sinh viên: Lê Đắc Tiến

Mã Sinh Viên: 20203100054

Lớp: ĐHTI14A1ND

1. Hàm kiểm tra chuỗi đầu vào

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **bool** **check\_ChuoiDL**(string chuoiDL)  {  doDai = chuoiDL.length();  **if**(doDai >= ARRAY\_SIZE)  {  **return** false;  }  **else**  {  **return** true;  }  } |

* Nhận vào một chuỗi, gán giá trị cho biến doDai, kiểm tra độ dài của chuỗi nhập có thỏa mãn yêu cầu không.

1. Hàm kiểm tra dữ liệu của khóa

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **bool** **check\_Khoa**(string khoaS)  {  stringstream ss(khoaS);  ss >> khoa;  **if** (ss.fail() || !ss.eof()) {  **return** false;  }  **return** true;  } |

* Nhận vào một chuỗi sau đó chuyển đổi chuỗi sang dạng số nguyên và lưu vào biến khoa. Nếu không chuyển đổi được thì trả về false, ngược lại trả về true.

1. Hàm xử lý dữ liệu đầu vào

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | **bool** **xu\_ly\_du\_lieu**(string chuoiDL,**int** key)  {  **for**(**int** i = **0** ; i < doDai ; i++)  {  **bool** check = false;  **char** c = chuoiDL[i];    **for**(**int** j = **0** ; j < **97**; j++)  {  **if**(bangASCII[j] == c)  {  **if**(key == **1**)  {  banRo[i] = j;  }  **else**  {  banMa[i] = j;  }  check = true;  **break**;  }  }  **if**(check == false)  {  cout<<"LOI!Khong the xu ly ky tu: "<<c<<"**\n**";  **return** false;  }  }  **return** true;  } |

* Nhận vào 1 chuỗi dữ liệu và 1 giá trị key để xác định xử lý dữ liệu cho việc giải mã hay mã hóa.
  + Duyệt qua tất cả các ký tự trong chuỗi nhập.
  + Mỗi ký tự ta so sanh với từng ký tự trong bảng tham chiếu.
  + Nếu ký tự đó trùng với ký tự nào trong bảng tham chiếu thì lấy ra vị trí của ký tự đó trong bảng tham chiếu.
  + Nếu key = 1 thì giá trị đó gán vào mảng banRo[], còn ngược lại thì giá trị được gán vào mảng banMa[].
  + Nếu trong trường hợp không tìm được ký tự nào tương ứng trong bảng tham chiếu thì trả về lỗi tại ký tự đó và dừng vòng lặp.

1. Hàm xử lý thuật toán Caesar

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | **bool** **thuat\_toan\_caesar**(**int** key)  {  try  {  **for**(**int** i=**0**; i<doDai; i++)  {  **if**(key == **1**)  {  **int** tmp = banRo[i] + khoa;  **while**(tmp<**0**)  {  tmp+=**97**;  }  banMa[i] = tmp%**97**;  }  **else**  {  **int** tmp = (banMa[i] - khoa);  **while** (tmp < **0**)  {  tmp += **97**;  }  banRo[i] = (tmp % **97**);  }  }  **return** true;  }  **catch**(exception& e)  {  **return** false;  }  } |

* Hàm nhận vào giá trị key để xác định mã hóa hoặc giải mã.
* Nếu key = 1 hàm thực hiện ma hóa.
  + Duyệt qua lần lượt từng phần tử trong mảng banRo[i].
  + Sau đó gán giá trị cho banMa[i] bằng banRo[i] + khóa.
  + Trong trường hợp giá trị âm thì công thêm với 97( độ dài bảng tham chiếu) để được giá trị dương.
  + Cuối cùng chia lấy phần dư cho 97 để đảm bảo giá trị luôn nắm trong bảng tham chiếu.
* Nếu key != 1 thì thực hiên giải mã. Quá trình giải mã làm ngược lại sẽ tiến hành gán giá trị cho banRo[i] bằng banMa[i] – khóa.

1. Hàm xuất dữ liệu

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | string **hien\_thi\_dau\_ra**(**int** key)  {  string chuoiDauRa = "";  try  {  **for**(**int** i=**0**;i<doDai;i++)  {  **char** c;  **if**(key == **1**)  {  c = bangASCII[banMa[i]];  }  **else**  {  c = bangASCII[banRo[i]];  }  chuoiDauRa+=c;  }  **return** chuoiDauRa;  }**catch**(exception& e)  {  **return** "Loi khong the xu ly du lieu";  }  } |

* Hàm nhận vào giá trị key để xác định xuất dữ liệu từ banRo hay banMa.
* Nếu key = 1 thì xuất dữ liệu từ banMa, ngược lại xuất dữ liệu từ banRo.
  + Duyệt qua từng phần tử của mảng cần xuất dữ liệu.
  + Với mỗi giá trị trong mảng thì sẽ lấy ra ký tự tương ứng tại vị trị giá trị trong mảng đó của bảng tham chiếu.
  + Thêm ký tự đó vào chuỗi chuoiDauRa.
  + Cuối cùng trả về chuoiDauRa đo.
* Quá trình lấy dữ liệu khi key != 1 hoàn toàn tương tự. Chỉ khác là sẽ sử dụng giá trị trong mảng banMa thay vì banRo.
* Trong trường hợp lấy dữ liệu bị lỗi thì sẽ trả về 1 chuỗi lỗi.

1. Quy trình thực hiện thuật toán (Hàm Main)
   * Chương trình có 3 lựa chọn
     + 1. Mã Hóa
     + 2. Giải Mã
     + Lựa chọn một số khác để thoát

* Nếu nhập 1 dữ liệu ngoài số nguyên thì yêu cầu nhập lại
* Khi chọn Mã Hóa hoăc Giải Mã thì sẽ yêu cầu nhập 2 dữ liệu
  + Dữ liệu đầu vào
  + Khóa
* Sau đó chạy hàm kiểm tra dữ liệu. Nếu hàm trả về false thì dừng và thông báo lỗi. Và quay lại menu lựa chọn. Nếu dữ liệu đầu vào thỏa mãn thì tiếp tục kiểm tra khóa. Nếu khóa không thỏa mãn thì thông báo lỗi và quay lại menu. Ngược lại khóa thỏa mãn thì tiến hành xử lý chuỗi dữ liệu vừa nhập để gán giá trị cho mảng bản rõ hoặc mảng bản mã tùy vào lựa chọn. Nếu xử lý thành công thì tiến hành thuật toán CAESAR theo lựa chọn. Khi thuật toán chạy thành công thì gọi hàm hiển thị dữ liệu và hiển thị dữ liệu ra tùy theo lựa chọn.
* Sau khi xử lý xong 1 vòng thì quay lại menu tiếp tục xử lý vòng lặp mới cho đến khi lựa chọn khác ngoài 1 và 2.

<https://github.com/LeTienIT/Caesar---Cpp---LeTienIT>