Nội dung

* Xâu ký tự
* Con trỏ và quản lý bộ nhớ cấp phát động

====================

Xâu ký tự trong C

* Là mảng các ký tự, thêm 1 ký tự báo hiệu kết thúc xâu là ‘\0’

char str[MAX];

Xâu str có thể chứa tối đa MAX-1 ký tự (vì mất 1 cho ký tự kết thúc)

* Truy cập các phần tử của xâu như là với mảng
* Nhập và dùng scanf(“%s”) --> không nhập được xâu có dấu cách trống
* Nhập dùng gets
* Dùng các hàm có sẵn trong thư viện string.h để xử lý xâu
  + strlen
  + strcmp
  + strcpy, strncpy
  + strcat
  + strchr
  + Strstr

Ví dụ. Nhập vào họ tên 1 người, tách ra phần tên và in ra lời chào

“Joe Biden” --> Biden  
“Elon” --> “Elon”  
C1. Tìm dấu cách trống cuối và in ra các ký tự từ sau dấu cách trống

|  |
| --- |
| #include<string.h>  #include<stdio.h>  int main()  {  char hoten[31];  printf("Ho ten: ");  gets(hoten);  int pos=-1;  for(pos=strlen(hoten); pos>=0; pos--)  if(hoten[pos]==' ') break;  printf("Xin chao ");  for(int i=pos+1; i<strlen(hoten); i++)  putchar(hoten[i]);  printf("\n");  return 0;  } |

C2. Tách riêng phần tên khỏi họ tên (nếu muốn dùng tên nhiều lần )

|  |
| --- |
| #include<string.h>  #include<stdio.h>  int main()  {  char hoten[31];  printf("Ho ten: ");  gets(hoten);  int pos=-1;  for(pos=strlen(hoten); pos>=0; pos--)  if(hoten[pos]==' ') break;  char ten[31];  int i;  // cach 1: Copy tung ky tu sang bien ten  for(i=pos+1; i<strlen(hoten); i++)  ten[i-(pos+1)] = hoten[i];  ten[i-(pos+1)] = '\0'; // bao hieu ket thuc xau  printf("Xin chao %s\n",ten);    return 0;  } |
| Cach 2:  //Cach 2: Dung strcpy  strcpy(ten, hoten+pos+1); //strcpy(ten, &hoten[pos+1]);  printf("Xin chao %s\n",ten); |

**Bài tập 1**. Cho danh sách họ tên sinh viên trong lớp (lớp tối đa 45 sinh viên)

Hãy viết chương trình nhập vào họ tên các thành viên trong lớp

Sắp xếp và in ra danh sách theo thứ tự TÊN được sắp theo ABC

VD. Nguyen Van Hung, Le Thi Anh, Tran Quang Dung thì thứ tự sau khi sắp sẽ là

Lê Thi Anh --> Tran Quang Dung --> Nguyen Van Hung

Gợi ý:

* Dùng mảng xâu ký tự dạng DS[45][31];
* Nhập vào từng họ tên dùng lệnh gets(DS[i])
* So sánh
  + Tách ra tên và so sánh dùng hàm strcmp()
  + Nên viết thành hàm so sánh
  + So sánh trực tiếp tên mà KHÔNG cần copy ra biến riêng?

|  |
| --- |
| #include<string.h>  #include<stdio.h>    int compareName(char \*hoten1, char \*hoten2)  {  int pos1;  for(pos1=strlen(hoten1)-1; pos1>=0; pos1--)  if(hoten1[pos1]==' ') break;  int pos2;  for(pos2=strlen(hoten2)-1; pos2>=0; pos2--)  if(hoten2[pos2]==' ') break;  return strcmp(hoten1+pos1+1, hoten2+pos2+1);  }  int main()  {  char hoten1[31]="Joe Biden", hoten2[31]="Elon Ausk";  printf("%d\n",compareName(hoten1,hoten2));  return 0;  } |

Sắp xếp danh sách

|  |
| --- |
| void sapxep(char DS[45][31], int n)  {  char tmp[31];  for(int i=n; i>1; i--)  for(int j=1; j<i; j++)  if(strcmp(DS[j-1],DS[j])>0)  {  strcpy(tmp,DS[j-1]);  strcpy(DS[j-1],DS[j]);  strcpy(DS[j],tmp);  }  } |

**Bài tập 2**. Xây dựng ứng dụng từ điển dạng đơn giản

File txt chứa từ điển EN trong đó từ trên 1 dòng (khoảng 371000 từ và độ dài mỗi từ ko quá 30 ký tự)



Hãy xây dựng chương trình đọc vào từ điển và kiểm tra các từ tiếng anh nhập vào từ bàn phím xem có đúng chính tả hay không

Link file: [words\_alpha.zip](https://husteduvn.sharepoint.com/:u:/s/20221C-BasicIT3230135310/EUAb57Qh4fhIrv84gCMVqAkBIKhDi0tYflt_Z7UoAxVr_g?e=riSmmX)

Chú ý. Với mảng kích thước lớn nên dùng cấp phát động bộ nhớ

=======================

**Con trỏ**

Với biến thông thường

|  |
| --- |
| #include<stdio.h>  int main()  {  char c1;  int a1, a2;  float f1;  double f2;  printf("Dia cho o nho cap phat cho c1: %#X\n",&c1);  printf("Dia cho o nho cap phat cho a1: %#X\n",&a1);  printf("Dia cho o nho cap phat cho a2: %#X\n",&a2);  printf("Kich thuoc vung nho cua c1: %d\n",sizeof(c1));  printf("Kich thuoc vung nho cua a1: %d\n",sizeof(a1));  printf("Kich thuoc vung nho cua a2: %d\n",sizeof(a2));  c1 = 'C';  printf("gia tri cua bien C1: %d\n",c1);  return 0;  } |

Biến được cấp phát 1 số lượng ô nhớ nào đó tùy theo kích thước

* Toán tử sizeof() = trả về kích thước/số ô nhớ cấp phát cho biến
* Địa chỉ ô nhớ đầu tiên cấp phát cho biến là toán từ &
* Tên biến = Giá trị mà đang được lưu trữ tại vùng nhớ của biến.
* Biến = đại diện cho 1 vùng nhớ nào đó trong máy tính

Con trỏ

* LÀ biến đặc biệt để lưu trữ địa chỉ ô nhớ
* Con trỏ khi được khai báo cũng được cấp phát địa chỉ ô nhớ
* Trên cùng 1 platform, số lượng bit để đánh địa chỉ ô nhớ là giống nhau --> Kích thước các con trỏ là giống nhau
* Kiểu dữ liệu khi khai báo con trỏ chỉ dùng để giới hạn phạm vi truy cập của con trỏ
* Chỉ có thể gán giá trị con trỏ theo 2 cách trong C/C++
  + Gán trực tiếp thông qua địa chỉ 1 biến cùng kiểu
  + Gán thông qua giá trị 1 con trỏ cùng kiểu
* TRuy cập giá trị vùng nhớ mà con trỏ đang trỏ tới thông qua toán tử \*
* Khi mới được khai báo, chưa được gán giá trị, con trỏ sẽ nhận mặc định hằng số NULL (chỉ ra con trỏ chưa được trỏ vào đâu cả)
* CẦN Phân biệt: Địa chỉ bộ nhớ cấp phát cho con trỏ VS Địa chỉ vùng nhớ mà con trỏ đang trỏ tới (giá trị của con trỏ)

|  |
| --- |
| #include<stdio.h>  int main()  {  char c1, \*pCh;  int a1, \*pint, \*p2;  double x1, \*pdb;  printf("dia chi vung nho cua pCh : %#X\n",&pCh);  printf("dia chi vung nho cua pint : %#X\n",&pint);  printf("dia chi vung nho cua p2 : %#X\n",&p2);  printf("dia chi vung nho cua pdb : %#X\n",&pdb);  printf("Kich thuoc cua bien pCh : %d\n",sizeof(pCh));  printf("Kich thuoc cua bien pint : %d\n",sizeof(pint));  printf("Kich thuoc cua bien p2 : %d\n",sizeof(p2));  printf("Kich thuoc cua bien pdb : %d\n",sizeof(pdb));  pint = &a1;  a1= 5;  printf("dia chi vung nho cua a1 : %#X\n",&a1);  printf("dia chi vung nho ma pint dang tro toi : %#X\n",pint);  printf("Gia tri vung nho ma pint dang tro toi : %d\n",\*pint);  \*pint = 1234;  printf("a1=%d\n",a1);  return 0;  } |

Con trỏ truy vập trường biến cấu trúc:

* Dùng (\*ptr).<tên trường>
* Hoặc ptr->Trường

**Con trỏ và hàm**

Truyền tham số cho hàm trong C la theo mô hình truyền bằng tham trị

* Tạo bản copy của tham số gốc trước khi gọi hàm
* Mọi thay đổi trong hàm chỉ tác động lên bản copy
* Khi kết thúc hàm, tham số gốc được phục hồi --> không chịu tác động của hàm

|  |
| --- |
| #include<stdio.h>  void swap1(int a, int b)  {  int c=a;  a=b;  b=c;  printf("Trong ham a=%d, b=%d\n",a,b);  }  int main()  {  int x=5, y=7;  printf("Truoc goi ham x=%d, y=%d\n",x,y);  swap1(x,y);  printf("Sau goi ham x=%d, y=%d\n",x,y);  return 0;  } |

Muốn thay đổi trong hàm tác động lên tham số gốc?

TRuyền vào địa chỉ tham số gốc thông qua con trỏ (mọi bản copy của địa chỉ đều trỏ tới 1 vùng nhớ duy nhất của tham số gốc)

|  |
| --- |
| #include<stdio.h>  void swap1(int a, int b)  {  int c=a;  a=b;  b=c;  printf("Trong ham a=%d, b=%d\n",a,b);  }    void swap2(int \*a, int \*b)  {  int c=\*a;  \*a=\*b;  \*b=c;  //printf("Trong ham a=%d, b=%d\n",a,b);  }  int main()  {  int x=5, y=7;  printf("Truoc goi ham x=%d, y=%d\n",x,y);  //swap1(x,y);  swap2(&x,&y);  printf("Sau goi ham x=%d, y=%d\n",x,y);  return 0;  } |