BÀI TẬP TUẦN 1

Môn học: Cấu Trúc Dữ Liệu và Giải Thuật.
GV hướng dẫn thực hành: Nguyễn Khánh Toàn (ktoan271199@gmail.com).
Nội dung chính:

- Ôn tập con trỏ, các thao tác trên mảng động.
- Ôn tập lại kiểu dữ liệu cấu trúc, cách đọc ghi file và một số thuật toán xử lý chuỗi.

Thời gian thực hiện: **2** tuần.

Bài tập được biên soạn lại và tham khảo từ tài liệu thực hành môn Cấu Trúc Dữ Liệu và Giải Thuật của bộ môn Công Nghệ Tri Thức, trường Đại Học Khoa Học Tự Nhiên TPHCM. Trân trọng cảm ơn quý Thầy Cô của bộ môn.

1 Con trỏ và mảng

1.1 Ôn tập kiến thức về con trỏ

Sinh viên thảo luận và trả lời một số câu hỏi sau:

Câu hỏi 1

Dự đoán kết quả của đoạn chương trình sau và giải thích.

```
# include <stdio.h>
void fun(int x)
{
    x = 30;
}

void fun2(int *x)
{
    *x = 30;
}

int main(){
  int y = 20;
  fun(y);
```

```
  printf("%d", y);
  fun2(&y);
  printf("%d", y);
  return 0;
}
```

Câu hỏi 2

Giả sử kiểu dữ liệu **float** có kích thước **4 bytes**, dự đoán kết quả của chương trình sau và giải thích kết quả dự đoán.

```
#include <stdio.h>

int main()
{
    float arr[5] = {12.5, 10.0, 13.5, 90.5, 0.5};
    float *ptr1 = &arr[0];
    float *ptr2 = ptr1 + 3;

    printf("%f", *ptr2);
    printf("%d", ptr2 - ptr1);

   return 0;
}
```

Câu hỏi 3

Giả sử rằng kiểu dữ liệu **int** có kích thước là **4 bytes**, kiểu **char** có kích thước là **1 bytes** và kích thước của **con trỏ** là **4 bytes**. Dự đoán kết quả của đoạn chương trình sau:

```
#include <stdio.h>

int main()
{
    int arri[] = {1, 2 ,3};
    int *ptri = arri;

    char arrc[] = {1, 2 ,3};
    char *ptrc = arrc;

    printf("%d", sizeof(arri));
    printf("%d", sizeof(ptri));

    printf("%d", sizeof(arrc));
    printf("%d", sizeof(ptrc));

    return 0;
}
```

Câu hỏi 4

Giả sử rằng kiểu dữ liệu **int** có kích thước là **4 bytes** và kiểu dữ liệu **char** có kích thước là **1 byte**. Dự đoán kết quả của chương trình dưới đây và giải thích.

```
#include<stdio.h>
int main()
```

```
{
    int arr[] = {10, 20, 30, 40, 50, 60};
    int *ptr1 = arr;
    int *ptr2 = arr + 5;
    printf("%d", (ptr2 - ptr1));
    printf("%d",  (char*)ptr2 - (char*) ptr1);
    return 0;
}
```

Câu hỏi 5

Cho biết kết quả của đoạn chương trình sau:

```
#include<stdio.h>
int f(int x, int *py, int **ppz)
{
  int y, z;
  **ppz += 1;
   z  = **ppz;
  *py += 2;
   y = *py;
   x += 3;
   return x + y + z;
}

void main()
{
   int c, *b, **a;
   c = 4;
   b = &c;
   a = &b;
   printf("%d", f(c, b, a));
   return 0;
}
```

Câu hỏi 6

Cho biết kết quả của đoạn chương trình sau:

```
#include<stdio.h>

void swap (char *x, char *y)
{
    char *t = x;
    x = y;
    y = t;
}

int main()
{
    char *x = "geeksquiz";
    char *y = "geeksforgeeks";
    char *t;
    swap(x, y);
    printf("(%s, %s)", x, y);
    t = x;
    x = y;
    y = t;
    printf("n(%s, %s)", x, y);
```

```
    return 0;
}
```

1.2 Mảng động với con trỏ

Mảng động là cấu trúc dữ liệu rất thuận tiện trong ngôn ngữ C, C++. Nguyên tắc hoạt động của mảng động như sau:

Ban đầu mảng động sẽ được xây dựng bằng cách khai báo một lượng bộ nhớ nhất định (thông thường lớn hơn số phần tử cần sử dụng). Các phần tử dữ liệu sẽ lần lượt được thêm vào mảng cho tới khi sử dụng hết lượng bộ nhớ đã khai báo.

Mảng động sẽ tự khai báo thêm bộ nhớ để lưu trữ dữ liệu trong trường hợp không đủ không gian để lưu trữ, thông thường là sẽ gấp đôi kích thước hiện có.

Sinh viên thực hiện cài đặt một số hàm số sau của mảng động:

a. Khởi tạo mảng động

int* initialize(**int** maxSize);

b. Thêm phần tử mới vào sau mảng động

void addElement(**int*** &array, **int** data, **int** &size, **int** &maxSize);

Lưu ý rằng cần thực hiện thêm hàm: **int*** growSize (**int*** array, **int** size, **int** &maxSize) trong quá trình thêm phần tử vào mảng động trong trường hợp mảng đầy.

c. Thêm phần tử tại một vị trí xác định vào mảng động

void addElementAtIndex(**int*** &array, **int** index, **int** data, **int** &size, **int** &maxSize);

d. Xóa phần tử cuối cùng của mảng động

void removeElement(**int*** array, **int** &size, **int** maxSize);

e. Xóa phần tử tại một vị trí xác định của mảng động

void removeElementAtIndex(**int*** array, **int** index, **int**&size, **int** maxSize);

f. Thu nhỏ kích thước mảng động: Thu nhỏ kích thước lưu trữ của mảng động xuống bằng với kích thước thực tế.

int* shrinkArray(**int*** array, **int** size, **int** &maxSize);

g. In các phần tử trong mảng động

void printArray(**int*** array, **int** size);

h. Tìm phần tử lớn nhất trong mảng động

int maxElement(**int*** array, **int** size);

k. Tìm kiếm một phần tử có giá trị xác định trong mảng động

int searchElement(**int*** array, **int** key, **int** size);

l. Cho hai mảng động được sắp xếp **tăng dần** với các phần tử phân biệt nhau. Viết hàm nối hai mảng động này sao cho mảng mới vẫn giữ được thứ tự tăng dần (sinh viên tự khai báo số lượng dữ liệu cần thiết cho mảng mới).

int* merge2Arrays(**int*** a, **int*** b, **int** na, **int** nb);

Sau khi thực hiện hoàn tất bài tập, các bạn sinh viên có thể tìm hiểu, tham khảo và sử dụng thư viện **std::<vector>** của ngôn ngữ C++.

1.3 Ôn tập kiểu dữ liệu cấu trúc, cách đọc ghi tập tin và các thuật toán xử lý chuỗi

Tập tin dữ liệu điểm thi THPT Quốc Gia của một số thí sinh năm 2018-2019 được lưu trữ trong file *"data.txt"*, bao gồm những nội dung như sau:

```
Số Báo Danh, Họ và Tên, Toán, Ngữ Văn, Vật Lý, Hóa Học, Sinh Học, Lịch Sử,
BD1200001,a van nguyen,4.0,5.0,,,,4.25,7.0,7.75,,,2.0,N1,BinhDinh
BD1200002,nhon hoai Le,7.0,6.25,6.0,6.25,6.5,,,,,,5.2,N1,BinhDinh
BD1200003,nam hoai Pham,5.2,5.75,,,,5.75,7.25,9.25,,,4.6,N1,BinhDinh
BD1200004,duc van Le,7.6,6.25,7.0,6.5,4.5,,,,,,6.2,N1,BinhDinh
BD1200005,phuong Cong Nguyen,8.6,6.5,4.0,7.25,5.5,,,,,,8.4,N1,BinhDinh
```

Hình 1: Demo tập tin

Giải thích dữ liệu:

- Dòng đầu tiên mô tả thông tin của các trường dữ liệu.
- Các dòng tiếp theo mô tả thông tin của từng thí sinh, mỗi thông tin cách nhau bởi một dấu phẩy.

- Các trường dữ liệu trống sẽ không có thông tin, nếu trường dữ liệu bị trống là điểm của một môn học, thì tương ứng với điểm môn học đó bằng 0.
- Các trường điểm thi KHTN và KHXH sẽ được mô tả ở phần tiếp theo.

Yêu cầu bài tập:

i. Thiết kế kiểu dữ liệu cấu trúc để lưu thông tin cho mỗi thí sinh.

```
struct thiSinh
{
    char *id, *hoVaTen;
    float toan, van, vatLy, hoaHoc, sinhHoc, lichSu, diaLy, giaoducCongDan, khoaHocTuNhien,
    khoaHocXaHoi, ngoaiNgu;
};
```

Hình 2: Demo cấu trúc thí sinh

ii. Cài đặt hàm số đọc thông tin của một thí sinh.

thiSinh docThongTinThiSinh(**char*** tenFile);

iii. Cài đặt hàm số đọc thông tin của một danh sách cách thí sinh.

vector<**thiSinh**> docDanhSachThiSinh(**char*** tenFile);

iv. Trong lúc làm thống kê điểm thi, do sai sót nên người thống kê viết tên thí sinh bị đảo ngược, ví dụ với thí sinh có số báo danh là *BD1200001*, họ và tên của thí sinh là *'nguyen van a'* tuy nhiên lại bị ghi ngược thành *'a van nguyen'*.

Các bạn sinh viên cài đặt hàm để đảo ngược lại tên của các thí sinh cho đúng thứ tự họ và tên.

void daoNguocTenThiSinh(**vector**<**thiSinh**> &danhSachThiSinh);

v. Viết hàm viết hoa các chữ cái đầu họ và tên của các thí sinh.

void vietHoaTenThiSinh(**vector**<**thiSinh**> &danhSachThiSinh);

vi. Do sai sót trong quá trình thống kê, các cặp thí sinh sau bị nhầm lẫn họ và tên với nhau:

- Thí sinh có SBD BD1200001 bị nhầm họ và tên với thí sinh BD1200003.

- Thí sinh có SBD BD1200005 bị nhầm họ và tên với thí sinh BD1200002.

Viết hàm hoán đổi họ và tên của các cặp thí sinh trên.

void doiTenThiSinh(**vector**<**thiSinh**> &danhSachThiSinh);

Lưu ý để mã nguồn được linh động hơn, các bạn nên viết thêm hàm số hoán đổi tên của hai thí sinh bất kỳ rồi sử dụng hàm này trong hàm trên. Tương tự với các câu trên, các bạn hãy chủ động viết thêm hàm xử lý tương ứng cho mỗi thí sinh, ví dụ hàm đảo ngược họ và tên cho mỗi thí sinh.

vii. Tính điểm thi.

Các bạn viết hàm tính điểm của khối thi Khoa học tự nhiên, khối thi khoa học xã hội và tổng điểm cho các thí sinh. Các điểm số này được tính như sau:

- Điểm KHTN = Điểm Toán + Điểm Vật Lý + Điểm Hóa Học.
- Điểm KHXH = Điểm Lịch Sử + Điểm Địa Lý + Điểm Giáo Dục Công Dân.
- Điểm Tổng = Điểm Toán + Điểm Văn + Điểm Ngoại Ngữ + Điểm KHTN + Điểm KHXH.

void tinhDiem(**vector**<**thiSinh**> &danhSachThiSinh);

Cũng giống như yêu cầu trước, để mã nguồn được linh động, các bạn nên có hàm tính tổng điểm từng khối thi cho mỗi thí sinh.

viii. Xuất file thống kê hoàn chỉnh.

Sau khi hoàn tất các yêu cầu trên, các bạn viết hàm xuất ra file *'tongketdiemthi.txt'* thống kê hoàn chỉnh với các trường thông tin sau:

- Số báo danh
- Họ và tên đã được thực hiện qua các hàm trước
- Điểm thi của khối thi khoa học tự nhiên
- Điểm thi của khối thi khoa học xã hội
- Điểm tổng

void xuatFile(**const vector**<**thiSinh**> &danhSachThiSinh, char* tenFile);

Sau khi thực hiện hoàn tất bài tập, các bạn sinh viên có thể tìm hiểu, tham khảo và sử dụng thư viện **std::<vector>**, **std::<string>** của ngôn ngữ C++.

Hết