Bài thực hành số 02. KIỂU DỮ LIỆU: XÂU KÝ TỰ, TỆP VÀ KIỂU DO NGƯỜI DÙNG ĐỊNH NGHĨA

Mục tiêu: Sử dụng được các kiểu dữ liệu xâu ký tự, tệp để lưu dữ liệu ra bộ nhớ ngoài và kiểu do người dùng định nghĩa để định nghĩa các kiểu dữ liệu không có trong các ngôn ngữ lập trình.

Nhắc lại kiến thức:

1. Kiểu xâu ký tự

Trong C++ kiểu xâu ký tự được định nghĩa trong std::string.

Ví dụ khai báo biến xâu ký tự s:

```
std::string s;
```

Hoặc khai báo không gian tên:

```
using namespace std;
string s;
```

+ Nhập/xuất xâu: để nhập xuất biến xâu có thể dùng cin, cout như các biến đơn giản. Tuy nhiên nếu nhập xâu bằng cin thì chỉ nhập được các xâu không có ký tự trống ở giữa.

Để nhập xâu có ký tự trống có thể dùng hàm getline:

```
getline(cin, s);
```

+ Gán xâu ký tự:

```
biến = “hằng xâu”;
```

Ví dụ: `s = “hello”;`

+ Thao tác với từng ký tự trong xâu:

```
biến[vị trí]
```

Ví dụ: `s[0] = ‘k’;`

+ Các phép toán so sánh: ==, !=, >, >=, <, <=

+ Một số phương thức trên xâu:

Tên phương thức	Ý nghĩa	Ví dụ
`size()`	Chiều dài xâu	`cout << s.size();`
`empty()`	Kiểm tra xâu rỗng	
`substr(n, m)`	Trả về xâu con từ ký tự thứ n, lấy m ký tự	`s = “Hello World”;`

		ss = s.sbustr(6, 5); // ss = “World”
find(str [,n])	Tìm xâu str trong xâu cho trước từ ký tự thứ n. Kết quả trả về vị trí tìm được hoặc -1	k = s.find(“ll”); // k = 2

2. Kiểu do người dùng định nghĩa (kiểu struct)

Kiểu struct dùng để định nghĩa ra những kiểu dữ liệu mới gồm nhiều kiểu dữ liệu kết hợp với nhau. Ví dụ kiểu PhanSo, SinhVien, Diem2D,…

Cú pháp khai báo kiểu cấu trúc:

```
struct TenKieu
{
	Kieu1 tenTruong1;
	…
	KieuN tenTruongN;
};
```

Ví dụ: Khai báo kiểu Student

```
struct Student
{
	string	name;
	int	age;
	double	gpa;
};
```

Khai báo biến kiểu cấu trúc:

```
	struct TenKieu tenBien;
```

Ví dụ:

```
	struct Student s;
```

Thao tác với các trường:

```
	tenBien.tenTruong
```

Ví dụ:

```
	s.gpa = 7.5;
```

** Nếu biến có kiểu con trỏ kiểu cấu trúc thì dùng cú pháp:

```
	tenBien->tenTruong
```

Phép gán:

```
tenBien2 = tenBien1;
```

Nhập/xuất: Nhập xuất từng trường.

3. Kiểu tệp

Kiểu tệp dùng để lưu dữ liệu của chương trình ra bộ nhớ ngoài và ngược lại.

Có 2 loại tệp:

+ Tệp văn bản: dữ liệu lưu dưới dạng các ký tự.

+ Tệp nhị phân: dữ liệu lưu dưới dạng dãy các byte.

Trong C++ có 3 lớp trong thư viện <fstream> dùng để thao tác với tệp:

`ifstream:` là lớp để đọc dữ liệu đầu vào từ tệp.

`ofstream:` là lớp để ghi dữ liệu ra tệp.

`fstream:` là lớp để đọc hoặc ghi dữ liệu từ tệp.

Các thao tác với tệp:

+ Mở tệp

+ Đọc/ghi dữ liệu tệp

+ Đóng tệp

Các phương thức:

+ Mở tệp:

```
open(const char *fileName, ios::mode)
```

`fileName`: tên tệp

`mode`: chế độ mở tệp

Chế độ	Ý nghĩa
`ios::app`	Chế độ Append. Dùng để ghi thêm vào cuối tệp
`ios::ate`	Mở một tệp để ghi và có thể di chuyển điều khiển read/write
`ios::in`	Mở tệp để đọc
`ios::out`	Mở tệp để ghi

Chế độ	Ý nghĩa
ios::trunc	Nếu tệp đã tồn tại, nội dung của nó sẽ được cắt (truncate) trước khi mở tệp
ios::binary	Thao tác với tệp nhị phân

+ Đọc tệp: với tệp văn bản có thể đọc, ghi như thao tác với bàn phím.

+ Đóng tệp: close()

Ví dụ: chương trình đọc xâu từ tệp input.txt và ghi xâu ra tệp output.txt.

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    fstream input("input.txt");
    if (!input)
    {
       cout << "File input.txt not found.";
       return 1;
    }
    fstream output;
    output.open("output.txt", ios::out);
    string str;
    input >> str; // đọc dữ liệu từ tệp vào biến str
    cout << str;
    output << "Happy new year 2023!!"; //ghi ra tệp output
    input.close(); //đóng tệp input
    output.close(); //đóng tệp output
    return 0;
}
```

** Lưu ý:

+ Để đọc 1 dòng trong tệp văn bản ta có thể dùng getline(fin, str). Khi đọc hết tệp hàm trả về giá trị false.

+ Phương thức eof() của fstream cho biết đọc dữ liệu đến hết tệp chưa.

+ Đọc/ghi tệp nhị phân:

write(char* data, int size): ghi dữ liệu từ địa chỉ của data với số byte dữ liệu cần ghi là size.

read(char* data, int size): đọc dữ liệu từ tệp size byte vào bộ nhớ tại địa chỉ của biến data.

Ví dụ: chương trình ghi một mảng các sinh viên ra tệp nhị phân students.bin.

```
#include <iostream>
#include <fstream>

using namespace std;

struct Student {
  char name[10];
  int age;
  float gpa;
};

int main() {
  // Create an array of Student structs
  Student students[3] = {{"Alice", 20, 3.9}, {"Bob", 21, 3.7},
{"Charlie", 22, 3.8}};

  // Open a binary file for writing
  ofstream out("students.bin", ios::binary);
  if (!out) {
    cerr << "Failed to open file for writing." << endl;
    return 1;
  }

  // Write the array of students to the file
  out.write((char*)students, sizeof(students));

  // Close the file
  out.close();

  return 0;
}
```

Chương trình đọc các sinh viên trong tệp students.bin

```
#include <iostream>
#include <fstream>
using namespace std;

struct Student {
  char name[10];
  int age;
  float gpa;
};

int main() {
  // Open the binary file for reading
  ifstream in("students.bin", ios::binary);
  if (!in) {
    cerr << "Failed to open file for reading." << endl;
    return 1;
  }

  // Read the array of students from the file
  Student student;
  while (in.read((char*)&student, sizeof(Student))) {
    cout << "Name: " << student.name << endl;
    cout << "Age: " << student.age << endl;
    cout << "GPA: " << student.gpa << endl;
  }

  // Close the file
  in.close();

  return 0;
}
```

Nội dung thực hành:

Bài 1. Danh sách sinh viên

Khai báo kiểu dữ liệu và viết các hàm quản lý một danh sách sinh viên gồm các thông tin: họ tên, tuổi, điểm trung bình.

Chương trình `Students.cpp`

```
#include <iostream>
using namespace std;
// define the data type of student
struct Student
{
  string   name;
  int      age;
  double   gpa;
};
// input a student
void inputStudent(struct Student &s)
{
  cout << "Name: "; getline(cin, s.name);
  cout << "Age: "; cin >> s.age;
  cout << "GPA: "; cin >> s.gpa;
}
// print a student
void printStudent(struct Student s)
{
  cout << "Name: " << s.name << endl;
  cout << "Age: " << s.age << endl;
  cout << "GPA: " << s.gpa << endl;
}
// input an array of students
void inputListOfStudent(struct Student list[], int n)
{
  int i;
  for(i = 0; i < n; i++)
  {
    cin.ignore();
    inputStudent(list[i]);
```

```
  }
}
// print an array of students
void printListOfStudent(struct Student list[], int n)
{
  int i;
  for(i = 0; i < n; i++)
    printStudent(list[i]);
}
// find a student by name
int findStudent(struct Student list[], int n, string name)
{
  int i;
  for(i = 0; i < n; i++)
    if(list[i].name == name)
      return i;
  return -1;
}


int main()
{
  struct Student m[100];
  string name;
  int k;
  inputListOfStudent(m, 5);
  printListOfStudent(m, 5);
  cin.ignore();
  cout << "Input name for find: "; getline(cin, name);
  k = findStudent(m, 5, name);
   if (k == -1)
    cout << "Not found!";
  else
    printStudent(m[k]);
}
```

Ghi mảng sinh viên ra tệp văn bản:

```
void writeToFile(struct Student list[], int n, char* fname)
{
  fstream out;
  out.open(fname, ios::out);
  for(int i = 0; i < n; i++)
    out << list[i].name << ", " << list[i].age << ", " << list[i].gpa
<< endl;
  out.close();
}
```

Kết quả ghi ra tệp có dạng:

```
Thu, 18, 8.5
Linh, 20, 8
```

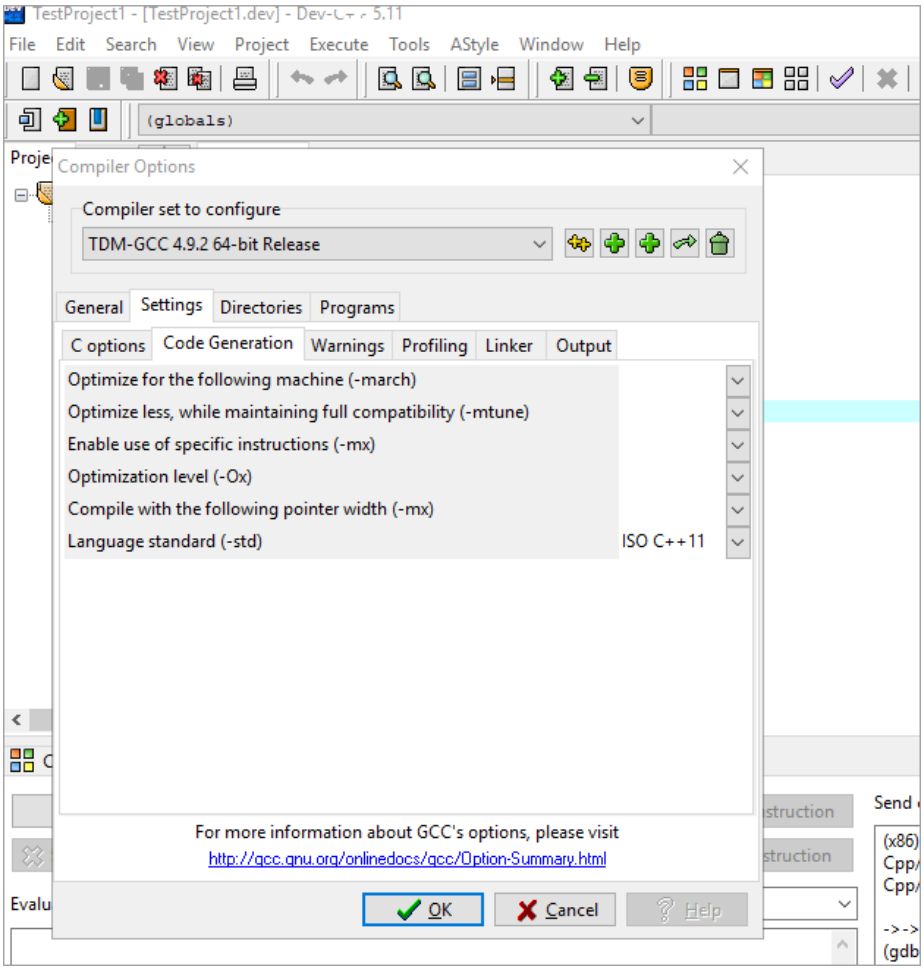
Đọc các sinh viên từ tập văn bản vào mảng:

```
int readFromFile(char *fname, struct Student list[])
{
  fstream fin;
  string s;
  int m, n, k = 0;
  fin.open(fname);
  while (getline(fin, s))
  {
    n = s.find(','); //find the character ‘,’ in s
    list[k].name = s.substr(0, n); // extract the name
    m = s.find(',', n+1); // character ‘,’ next in s
    list[k].age = stoi(s.substr(n+2, m-n-2)); //extract age
    list[k].gpa = stof(s.substr(m+2));  //extract gpa
    k++;
  }
  fin.close();
  return k;
}
```

Hãy bổ sung hai hàm trên vào chương trình `Students.cpp` và thêm những lệnh để ghi mảng các sinh viên nhập vào ra tệp `students.txt` sau đó đọc tệp này ra một mảng sinh viên và in mảng này để kiểm tra kết quả.

Lưu ý: các hàm stoi, stof chỉ hỗ trợ từ C++11 nên cần chuyển đổi sang bộ biên dịch bằng C++11, trong DevC++ thực hiện như sau:

- Chọn menu **Tools** trong Dev-C++ IDE.
- Chọn **Compiler Options...**
- Chọn tab "**Settings**"
- Chọn tab "**Code generation**"
- Click mục "**Language Standard (-std)**" chọn "**ISOC++11**" hoặc "**GNUC++11**".



Bổ sung: viết các hàm sau trên mảng các sinh viên, sau đó gọi hàm trong main để thực hiện chức năng của hàm.

a) Liệt kê các sinh viên có gpa>=8.

```
void listOfGoodStudents(struct Student list[], int n)
{
```

```
}
```

b) Thêm một sinh viên vào cuối mảng chứa n sinh viên

```
int appendStudent(struct Student list[], int n, struct Student s)
{



}
```

c) Đếm số người họ "Nguyen" trong mảng sinh viên.

```
int countFirstName(struct Student list[], int n)
{



}
```

d) Sắp xếp mảng sinh viên theo thứ tự tăng của họ tên kiểu người Việt.

Gợi ý: viết hàm chuyển họ tên viết theo kiểu người Việt: họ - đệm - tên thành dạng: tên họ đệm.

Ví dụ: "Nguyen Van Anh" chuyển thành "Anh Nguyen Van"

```
string convertName(string name)
{



}
```

Sử dụng hàm trên để sắp xếp:

```
void sortVNName(struct Student list[], int n)
{




}
```
