Bài thực hành 3

Mục tiêu

- Dùng kết hợp các lệnh điều khiển luồng: íf-else, switch, for, while, do-while
- Sử dụng một số hàm có sẵn trong thư viện
- Viết hàm của riêng bạn
- Hiểu phạm vi hoạt động của biến

Bạn không được dùng mảng trong bài tập tuần này.

Câu 1. [draw.cpp]

Viết chương trình vẽ hình theo mô tả dưới đây.

a) Viết hàm vẽ tam giác cân chiều cao h bằng các kí tự '*' sử dụng khai báo:

```
void drawTriangle(int h);
```

Ví dụ tam giác với h = 3:

```
    *
   ***
  *****
```

b) Viết hàm vẽ hình chữ nhật chiều rộng a, chiều dài b bằng các kí tự '*' sử dụng khai báo:

```
void drawRectangle(int a, int b);
```

Ví dụ hình chữ nhật với a = 4, b = 8

```
********
********
********
********
```

c) Viết hàm vẽ hình chữ nhật chiều rộng a, chiều dài b chỉ có các cạnh vẽ bằng kí tự '*' sử dụng khai báo:

```
void drawRectangleBorder(int a, int b);
```

Ví dụ hình chữ nhật với a = 4, b = 8

```
********
*      *
*      *
********
```

d) Viết hàm `main()` in ra menu cho phép người dùng chọn một trong các tính năng vẽ nói trên hoặc thoát chương trình. Sau khi người dùng lựa chọn, vẽ hình tương ứng và in lại menu.

Ví dụ menu

```
Chương trình vẽ hình đơn giản
* Vẽ tam giác cân: Bấm 1 rồi enter
* Vẽ hình chữ nhật: Bấm 2 rồi enter
* Vẽ hình chữ nhật rỗng: Bấm 3 rồi enter
* Thoát chương trình: Bấm 0 rồi enter
Lựa chọn của bạn: _
```

Câu 2. [uinteger.cpp]

Viết chương trình xử lý số nguyên dương theo mô tả dưới đây

a) Viết hàm tính tổng các chữ số của số n sử dụng khai báo

```
unsigned int sumDigit(unsigned int n);
```

Ví dụ n = 1992 thì hàm trả về giá trị 1 + 9 + 9 + 2 = 21

b) Viết hàm tìm số có các chữ số ngược với số n sử dụng khai báo

```
unsigned int reverse(unsigned int n);
```

Ví dụ n = 1992 thì hàm trả về giá trị 2991

c) Viết hàm `isPerfect` kiểm tra xem số n có phải số hoàn hảo hay không. Số n là hoàn hảo nếu nó bằng tổng các ước <n của nó.

Ví dụ: 6 = 1 + 2 + 3, 28 = 1 + 2 + 4 + 7 + 14 là các số hoàn hảo.

d) Viết hàm `isPrime` kiểm tra xem số n có phải số nguyên tố hay không. Số n là nguyên tố nếu n ≥ 2 và chỉ chia hết cho 1 và chính nó.

Ví dụ: 2, 17, 53 là số nguyên tố

e) Viết hàm `factor` phân tích n thành thừa số nguyên tố và in ra dạng đã phân tích.

Ví dụ: với n = 1998, hàm này in ra 1998 = 2 * 3 ^ 3 * 37

e) Viết hàm `main()` cho phép người dùng nhập vào một số nguyên dương n và in ra:

- tổng số chữ số của n
- số có các chữ số ngược với n
- n có phải số hoàn hảo không
- n có phải số nguyên tố không
- kết quả phân tích n thành thừa số nguyên tố.

Câu 3. [guess.cpp]

Viết chương trình cho phép người dùng đoán một số n ngẫu nhiên (trong khoảng từ 1 đến 100). Ta sinh n với hàm rand. Chú ý là n khác nhau ở các lần chạy khác nhau. Khi người dùng đoán giá trị x,

- nếu x bằng n, in ra "CHÍNH XÁC!" và kết thúc chương trình
- nếu x lớn hơn n, in ra "n < x" rồi lặp lại việc nhập và so sánh
- nếu x nhỏ hơn n, in ra "n > x" rồi lặp lại việc nhập và so sánh

Ví dụ:

```
Chương trình đoán số n (1 <= n <= 100).
n theo phỏng đoán của bạn: 25
n < 25
n theo phỏng đoán của bạn: 10
n < 10
n theo phỏng đoán của bạn: 8
n > 8
n theo phỏng đoán của bạn: 9
CHÍNH XÁC!
```