Информатика

Кочанов Марк

27 октября 2018 г.

МИФИ

Оператор switch

switch

```
1  char x = 2;
2
3  switch (x) {
4      case 1:
5          printf("1");
6          break;
7      case 2:
8          printf("2");
9          break;
10     default:
11         printf("value of x unknown");
12 }
```

- Для x допустим любой целочисленный тип (int, char, usigned int и тд)
- Для x допустим тип enum
- Ветвь default выполняется в случае, если не нашлось подходящего значения ни в одной из ветвей

switch

```
1  switch (i) {
2      case 1:
3      case 2:
4      case 3:
5          printf("1-3");
6          break;
7      case 4:
8          printf("4");
9      case 5:
10         printf("4-5");
11         break;
12 }
```

- Если в конце ветки отсутствует оператор break, то выполнится следующая ветвь
- Ветвь может быть пустой, либо содержать один оператор break

Тип данных enum

Перечисляемый тип данных enum

```
1  enum color {RED, GREEN, BLUE};
2  enum r = RED;
3  switch(r) {
4      case RED:
5          puts("red");
6          break;
7      case GREEN:
8          puts("green");
9          break;
10     case BLUE:
11         puts("blue");
12         break;
13 }
```

- При указании типа переменной необходимо использовать ключевое слово enum, либо использовать ключевое слово typedef
- Значение переменной типа enum представляет собой целое число

Перечисляемый тип данных enum

```
1  enum foo {A, B, C=10, D, E=1, F, G=F+C} f = D; // f = 11
2  enum color {RED, GREEN, BLUE} c1, c2;
3  c1 = GREEN; // c1 = 1
4
5  enum TV {FOX=11, CNN=25, ESPN=15, HBO=22, MAX=30, NBC=32};
6  printf("List of cable station : \n");
7  printf(" FOX: %d\n", FOX);
8  printf(" HBO: %d\n", HBO);
9  printf(" MAX: %d\n", MAX);
```

- Переменные данного типа могут быть созданы при объявлении типа
- При выводе при помощи print использовать те же модификаторы типов, что и при выводе целых чисел

Передача enum в функцию

```
1  enum color {RED, GREEN, BLUE};
2  typedef enum color color_t;
3  // the same as
4  // typedef enum color {RED, GREEN, BLUE} color_t;
5
6  color_t transform_color(color_t old_color) {
7      color_t c = (old_color + 1) % 3;
8      return c;
9  };
10
11 color_t c1 = RED;
12 color_t c2 = transform_color(c1);
13 color_t c3 = transform_color(c2);
14 color_t c4 = transform_color(c3);
15
16 printf("%u %u %u, %u\n", c1, c2, c3, c4); // 0 1 2 0
```

Использование enum для возврата статуса

```
1  enum status {POS, NEG, ZERO_DIV};
2  typedef enum status status_t;
3
4  status_t frac_create(int num, int denom, frac* res);
5
6  frac frac_1;
7  status_t s1 = frac_create(10, 20, &frac_1);
8
9  if (s1 == POS)
10     printf("Frac is created, > 0");
11 else if (s1 == NEG)
12     printf("Frac is created, < 0");
13 else
14     printf("Error, zero division");
```

Задача

Комплексные числа

Реализовать код для работы с комплексными числами, где вещественная и мнимая часть задача в виде рационального числа (дроби). Необходимо объявить стуктуру для хранения комплексного числа. Для задания вещественной и мнимой части использовать стуктуру, использованную в предыдущей лабораторной работе. Необходимо реализовать функция для сложения, вычитания, умножения двух комплексных чисел. Функция умножения должна возвращать статус (умножение прошло успешно, либо была попытка деления на ноль) через enum.