Указатель на функцию

Указатель на функцию

- Объявление указателя на функцию

```
double trapezium(double a, double b, int n,
                                 double (*func)(double));
```

- Получение адреса функции

```
result = trapezium(0, 3.14, 25, &sin /* sin */);
```

- Вызов функции по указателю

```
y = (*func)(x);    // y = func(x);
```

qsort (stdlib.h)

```
void qsort(void *base, size_t nmemb, size_t size,
           int (*compar)(const void*, const void*));
```

Пусть необходимо упорядочить массив целых чисел по возрастанию.

```
int compare_int(const void* p, const void* q)
{
    const int *a = p;
    const int *b = q;
    return *a - *b;    // return *(int*)p - *(int*)q;
}
...
int a[10];
...
qsort(a, sizeof(a) / sizeof(a[0]), sizeof(a[0]),
                                          compare_int);
```

Особенности использования указателей на функции (1)

Согласно C99 6.7.5.3 #8, выражение из имени функции неявно преобразуется в указатель на функцию.

```
int add(int a, int b);
...
int (*p1)(int, int) = add;
```

Операция "&" для функции возвращает указатель на функцию, но из-за 6.7.5.3 #8 это лишняя операция.

```
int (*p2)(int, int) = &add;
```

Особенности использования указателей на функции (2)

Операция "*" для указателя на функцию возвращает саму функцию, которая неявно преобразуется в указатель на функцию.

```
int (*p3)(int, int) = *add;
int (*p4)(int, int) = *****add;
```

Указатели на функции можно сравнивать

```
if (p1 == add)
    printf("p1 points to add\n");
```

Особенности использования указателей на функции (3)

Указатель на функцию может быть типом возвращаемого значения функции

```
int (*get_action(char ch))(int, int);

// typedef приходит на помощь :)
typedef int (*ptr_action_t)(int, int);

ptr_action_t get_action(char ch);
```

Указатель на функцию и void* (1)

C99 6.3.2.3 #1

A pointer to void may be converted to or from a pointer to any incomplete or object type. A pointer to any incomplete or object type may be converted to a pointer to void and back again; the result shall compare equal to the original pointer.

Функция - *не объект* в терминологии стандарта.

Указатель на функцию и void* (2)

C99 6.3.2.3 #8

A pointer to a function of one type may be converted to a pointer to a function of another type and back again; the result shall compare equal to the original pointer. If a converted pointer is used to call a function whose type is not compatible with the pointed-to type, the behavior is undefined.

Указатель на функцию и void* (3)

Согласно C99 6.3.2.3 #1 и C99 6.3.2.3 #8, указатель на функцию не может быть преобразован к указателю на void и наоборот.

Но POSIX требует, чтобы такое преобразование было возможно при работе с динамическими библиотеками.

- C99 J.5.7 Function pointer casts (расширение стандарта)
- POSIX dlsym RATIONALE
- Generic Function Pointer C2X (будущее (?))

Использование указателей на функции (1)

С помощью указателей на функции в языке Си реализуются

- функции обратного вызова (англ., callback);
- таблицы переходов (англ., jump table);
- динамическое связывание (англ., binding).

Использование указателей на функции (2)

Callback (англ, *функция обратного вызова*) - передача исполняемого кода в качестве одного из параметров другого кода. [wiki]

Функция обратного вызова - это "действие", передаваемое в функцию в качестве аргумента, которое обычно используется

- для обработки данных внутри функции (map);
- для того, чтобы «связываться» с тем, кто вызвал функции, при наступлении какого-то события.