Указатели типа void*

Теги: void, пустые указатели, неопределённые указатели.*

Указатели типа void

В си существует особый тип указателей – указатели типа void или пустые указатели. Эти указатели используются в том случае, когда тип переменной не известен. Так как void не имеет типа, то к нему не применима операция разадресации (взятие содержимого) и адресная арифметика, так как неизвестно представление данных. Тем не менее, если мы работаем с указателем типа void, то нам доступны операции сравнения.

Если необходимо работать с пустым указателем, то сначала нужно явно привести тип. Например

```
#include <conio.h>
#include <stdio.h>

int main() {
    void *p = NULL;
    int intVar     = 10;
    char charVar   = 'A';
    float floatVar = 24.3;
    float *floatPtr = NULL;

    p = &intVar;
    *((int*) p) = 20;
    printf("intVar = %d\n", intVar);

    p = &charVar;
    printf("charVar = %c\n", *((char*) p));

    p = &floatVar;
    floatPtr = (float*) p;
    printf("floatVar = %.3f", *floatPtr);

    getch();
}
```

Переменная не может иметь типа void, этот тип определён только для указателей. Пустые указатели нашли широкое применение при вызове функций. Можно написать функцию общего назначения, которая будет работать с любым типом. Например, напишем функцию swap, которая обменивает местами содержимое двух переменных. У этой функции будет три аргумента – указатели на переменные, которые необходимо обменять местами и их размер.

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void swap(void *a, void *b, size_t size) {
    char* tmp;
    //создаём временную переменную для обмена
    tmp = (char*) malloc(size);
    memcpy(tmp, a, size);
    memcpy(a, b, size);
    memcpy(b, tmp, size);
    free(tmp);
}

int main() {
    float a = 10.f;
    float b = 20.f;
    double c = 555;
    double d = 777;
    unsigned long e = 2ll;
    unsigned long f = 3ll;

```

```
    printf("a = %.3f, b = %.3f\n", a, b);
    swap(&a, &b, sizeof(float));
    printf("a = %.3f, b = %.3f\n", a, b);

    printf("c = %.3f, d = %.3f\n", c, d);
    swap(&c, &d, sizeof(double));
    printf("c = %.3f, d = %.3f\n", c, d);

    printf("e = %ld, f = %ld \n", e, f);
    swap(&e, &f, sizeof(unsigned long));
    printf("e = %ld, f = %ld \n", e, f);

    getch();
}
```

Наша функция может выглядеть и по-другому. Обойдёмся без дорогостоящего выделения памяти и будем копировать побайтно.

```
void swap(void *a, void *b, size_t size) {
    char tmp;
    size_t i;
    for (i = 0; i < size; i++) {
        tmp = *((char*) b + i);
        *((char*) b + i) = *((char*) a + i);
        *((char*) a + i) = tmp;
    }
}
```

Пустые указатели позволяют создавать функции, которые возвращают и принимают одинаковые параметры, но имеют разное название. Это пригодится в дальнейшем, при изучении указателей на функции. Например

```
int cmpInt(void* a, void* b) {
    return *((int*) a) - *((int*) b);
}

int cmpString(void *a, void* b) {
    return strcmp((char*) a, (char*) b);
}

int cmpFloat(void* a, void* b) {
    float fdiff = *((float*) a) - *((float*) b);
    if (fabs(fdiff) < 0.000001f) {
        return 0;
    }
    if (fdiff < 0) {
        return -1;
    } else {
        return 1;
    }
}
```

Реализация вызова функции

Указатели на функции