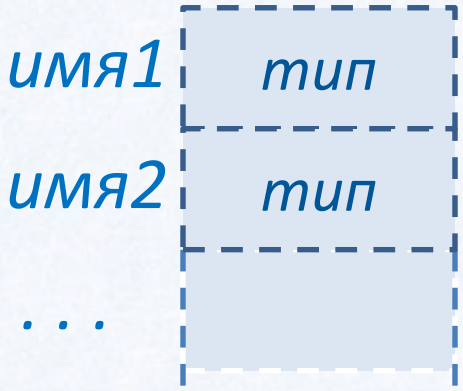
Материалы презентации предназначены для размещения только для использования студентами кафедры «Компьютерные системы и технологии» НИЯУ МИФИ дневного и вечернего отделений, изучающими курс «Программирование (Алгоритмы и структуры данных)».

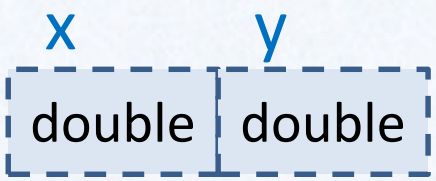
С4. Структуры

Определение структуры

```
struct имя_структуры {
    тип  имя1, … ;
    тип  имя2, … ;
  . . .
};
```



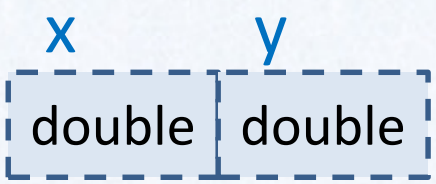

```
struct Point {
   double x, y;
};
```



Определение структуры

```
typedef struct имя_структуры {
    тип  имя1, … ;
    тип  имя2, … ;
  . . .
} новое_имя_типа;
```

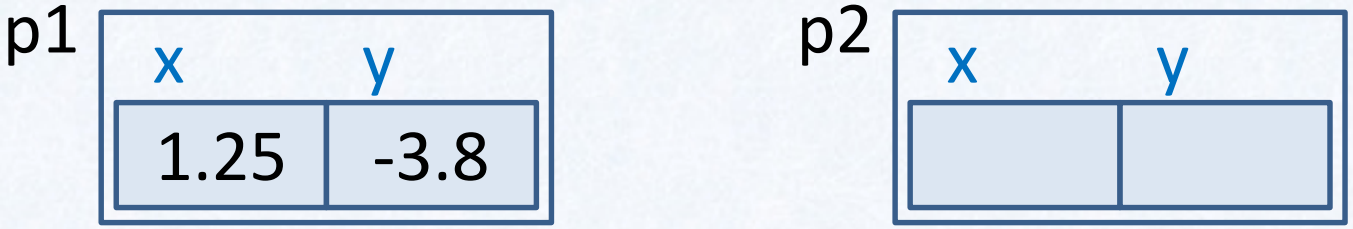
```
typedef struct Point {
   double x, y;
} Point;
```



Определение переменных

struct *имя_структуры имя_переменной*
[= {*значение_1*, *значение_2*, ... }] ;

struct Point p1 = {1.25, -3.8}, p2;



Определение переменных

Если использовался typedef:

имя_структуры имя_переменной

[= {*значение_1*, *значение_2*, ... }] ;

Point p1 = {1.25, -3.8}, p2;

Определение массива

struct *имя_структуры имя_массива*[*количество*]

[= { { значение_01, значение_02, . . . },

{ значение_11, значение_12, . . . },

. . .

}];

struct Point pp[3] =

{{1, 1}, {2,2}, {1,2}};

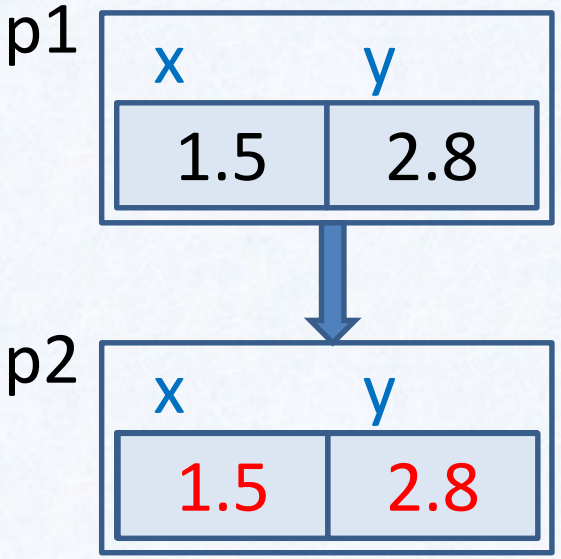
pp

x	y
1.0	1.0
2.0	2.0
1.0	2.0

Копирование структуры

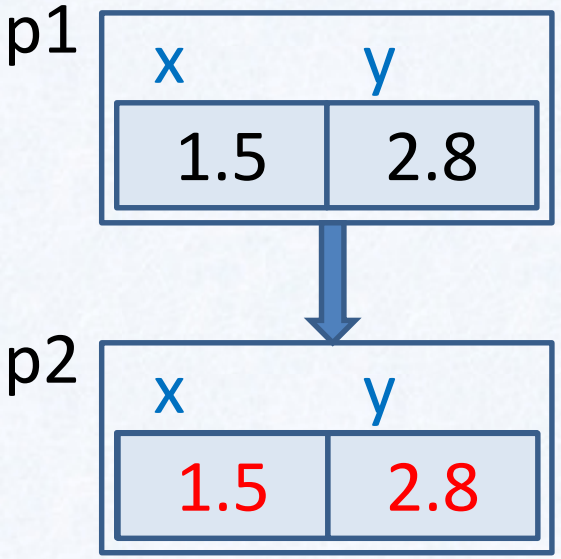
```
struct Point p1 = {1.5, 2.8};
```

```
struct Point p2 = p1;
```



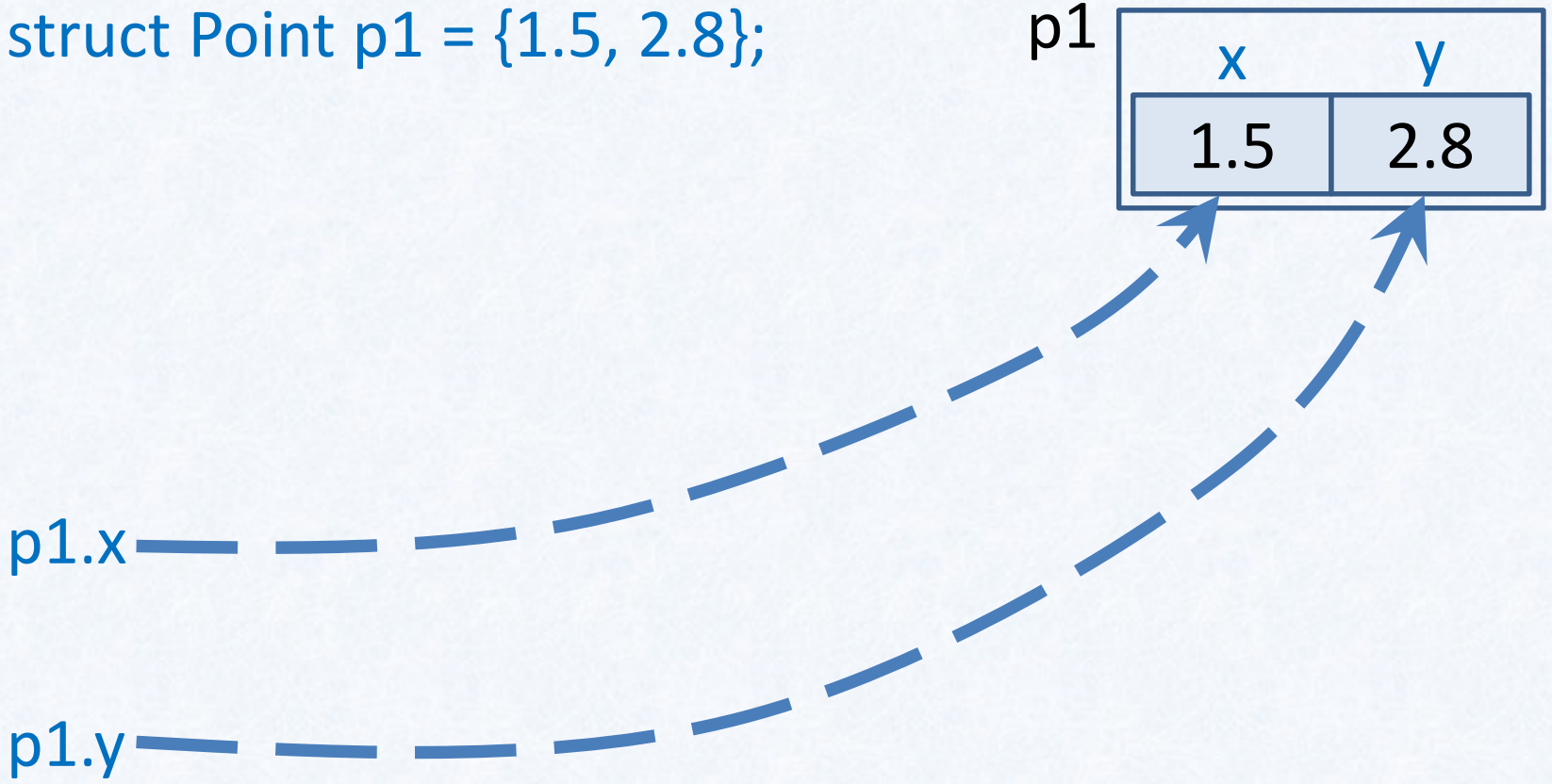
Присваивание структуры

```
struct Point p1 = {1.5, 2.8};

struct Point p3;
p3 = p1;
```



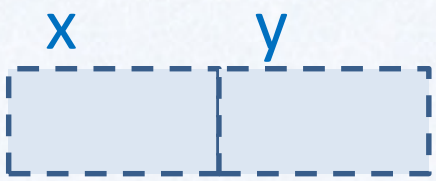
Разыменование структуры

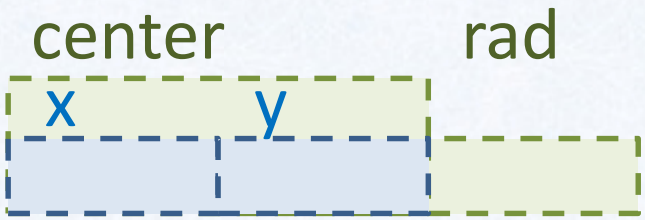
struct Point p1 = {1.5, 2.8};



Вложенные структуры

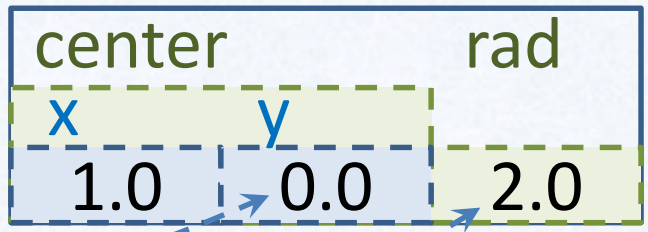
```
struct  Point {
    double x,y;
};

struct  Circle {
    struct Point  center;
    double  rad;
};
```





Вложенные структуры

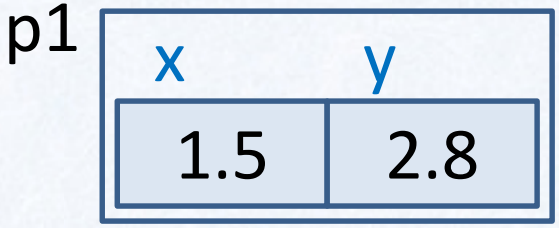
```
struct  Circle c = {{1, 0}, 2};
```




```
c.center.y
c.rad
```

Указатели на структуру

```
struct  Point {
    double x, y;
};
struct  Point  p1 = {1.5, 2.8};
struct  Point  *ptr;
```

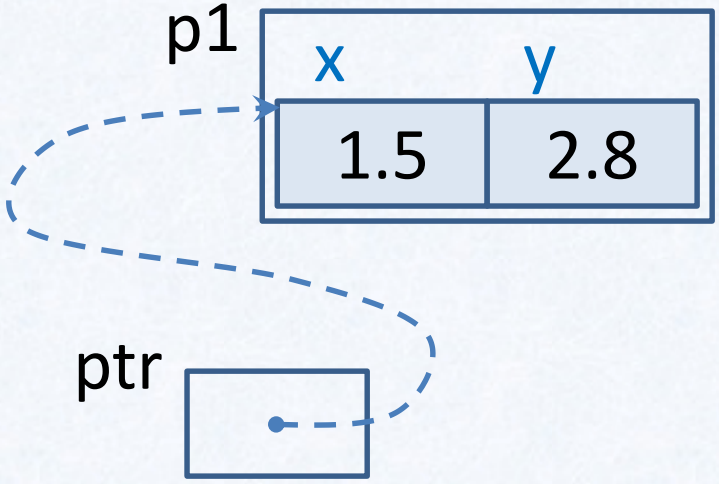


Указатели на структуру

```
struct  Point {
    double x, y;
};
struct  Point  p1 = {1.5, 2.8};
struct  Point  *ptr = &p1;
```

Разыменование структуры:

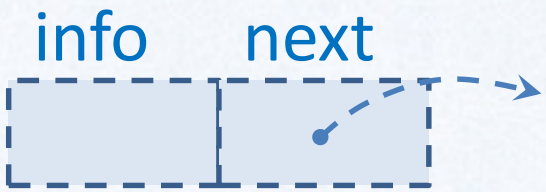
(*ptr).x ptr -> x



Списки

Элемент списка

```
struct  Item {
    тип  info;
    struct  Item *next;
};
```



Начало списка

```
struct  Item *first;
```

Примеры работы со списком

1. Создать список из символьной строки
   (или ввести строку-список из входного потока)
2. Вывести список в выходной поток
3. Освободить память, занятую списком

Объявления

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

typedef struct Item {
    char c;
    struct Item *next;
} Item;

Item *createList(const char *);
// или int getList (Item **);
void putList(Item *);
Item *deleteList(Item *);
```

Функция createList()

```
Item *createList(const char *str)
{
    Item head = {'*', NULL};
    Item *last = &head;
    while(*str != '\0'){
        last->next = (Item *)malloc(sizeof(Item));
        last = last->next;
        last->c = *str++;
        last->next = NULL;
    }
    return head.next;
}
```

Функция putList()

```
void putList(char *msg, Item *ptr)
{
    printf("%s: \"", msg);
    for(; ptr != NULL; ptr = ptr->next)
        printf("%c", ptr->c);
    printf("\"\n");
}
```

Функция deleteList()

```
Item *deleteList(Item *ptr)
{
    Item *tmp = NULL;
    while(ptr != NULL){
        tmp = ptr;
        ptr = ptr->next;
        free(tmp);
    }
    return ptr;
}
```

Тестирование

```
int main()
{
	char buf[80];
	Item *st;
	while(puts("enter string"), gets(buf)){
		st = createList(buf);
		putList("Entered string", st);
		st = deleteList(st);
	}
	return 0;
}
```

Функция getList()

```
int getList(Item **pptr)
{
    char buf[21], *str;
    Item head = {'*', NULL};
    Item *last = &head;
    int n, rc = 1;
    do{
        n = scanf("%20[^\n]", buf);
```

Функция getList()

```
if(n > 0){
    for(str = buf; *str != '\0'; ++str){
        last->next = (Item *)malloc(sizeof(Item));
        last = last->next;
        last->c = *str;
    }
    last->next = NULL;
    continue;
}
```

Функция getList()

```
        if(n < 0){
            deleteList(head.next);
            head.next = NULL;
            rc = 0;
        }
        else
            scanf("%*c");
    } while(n > 0);
    *pptr = head.next;
    return rc;
}
```

Тестирование

```
int main()
{
	Item *st;
	while(puts("enter string"), getList(&st)){
		putList("Entered string", st);
		st = deleteList(st);
	}
	return 0;
}
```