

U3 3. Übung

Aufgabe 2

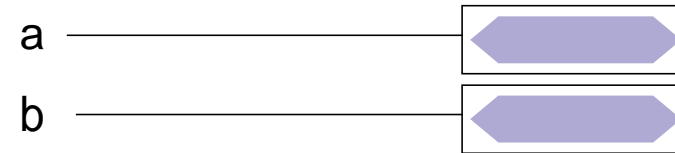
U3-1 einfache swap_double Funktion

- Parameter werden in C *by-value* übergeben
- die aufgerufene Funktion kann den aktuellen Parameter beim Aufrufer nicht verändern
- auch Zeiger werden *by-value* übergeben, d. h. die Funktion erhält lediglich eine Kopie des Adressverweises
- über diesen Verweis kann die Funktion jedoch mit Hilfe des *-Operators auf die zugehörige Variable zugreifen und sie verändern
 - ↳ *call-by-reference*

1 Zeiger als Funktionsargumente

■ Beispiel:

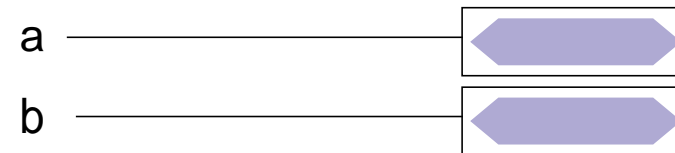
```
void swap (double *, double *);  
int main(void) {  
    double a, b;  
    ...  
    swap(&a, &b);  
}
```



1 Zeiger als Funktionsargumente

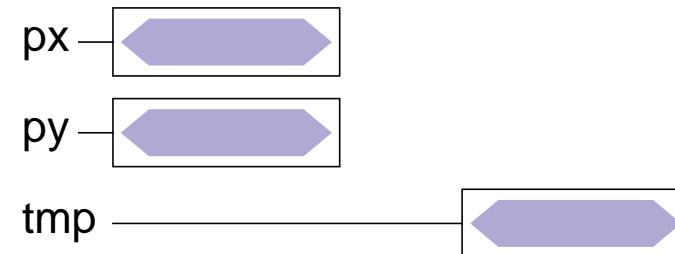
■ Beispiel:

```
void swap (double *, double *);
int main(void) {
double a, b;
...
swap(&a, &b);
}
```



```
void swap (double *px, double *py)
{
double tmp;

tmp = *px;
*px = *py;
*py = tmp;
}
```



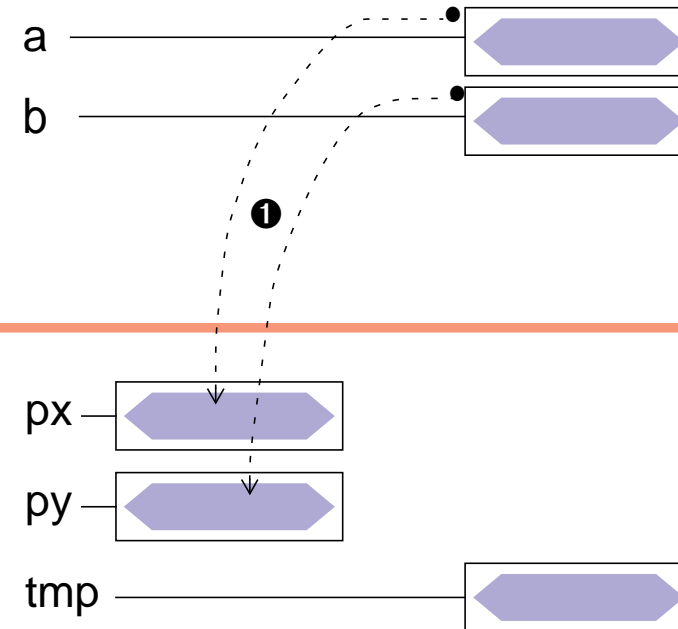
1 Zeiger als Funktionsargumente

■ Beispiel:

```
void swap (double *, double *);
int main(void) {
double a, b;
...
swap(&a, &b); ①
}
```

```
void swap (double *px, double *py)
{
double tmp;

tmp = *px;
*px = *py;
*py = tmp;
}
```



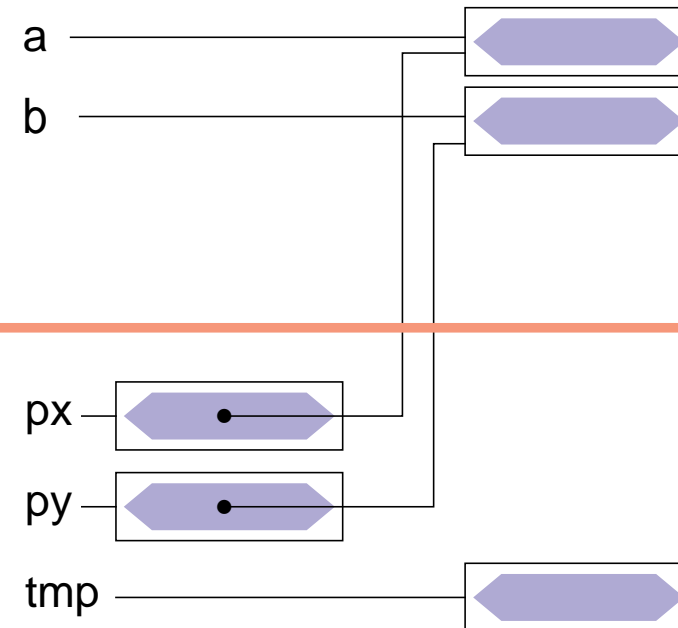
1 Zeiger als Funktionsargumente

■ Beispiel:

```
void swap (double *, double *);
int main(void) {
double a, b;
...
swap(&a, &b);
}
```

```
void swap (double *px, double *py)
{
double tmp;

tmp = *px;
*px = *py;
*py = tmp;
}
```



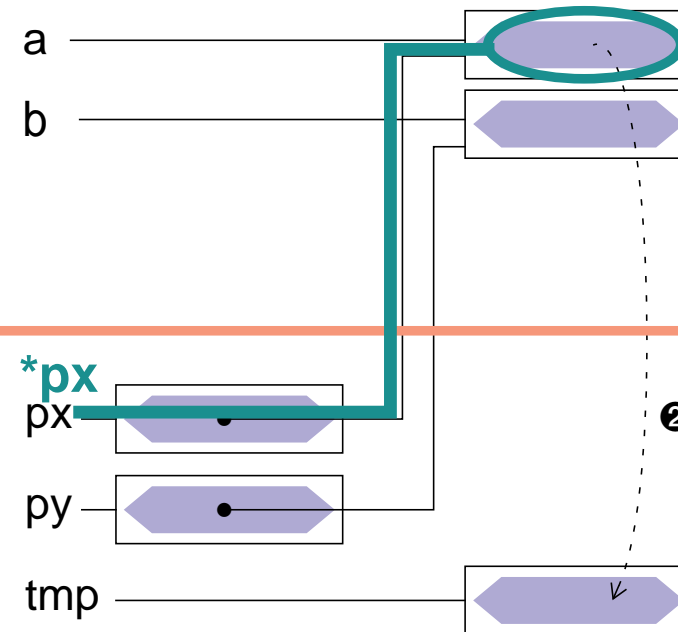
1 Zeiger als Funktionsargumente

■ Beispiel:

```
void swap (double *, double *);
int main(void) {
double a, b;
...
swap(&a, &b); ①
}
```

```
void swap (double *px, double *py)
{
double tmp;

tmp = *px; ②
*px = *py;
*py = tmp;
}
```



1 Zeiger als Funktionsargumente

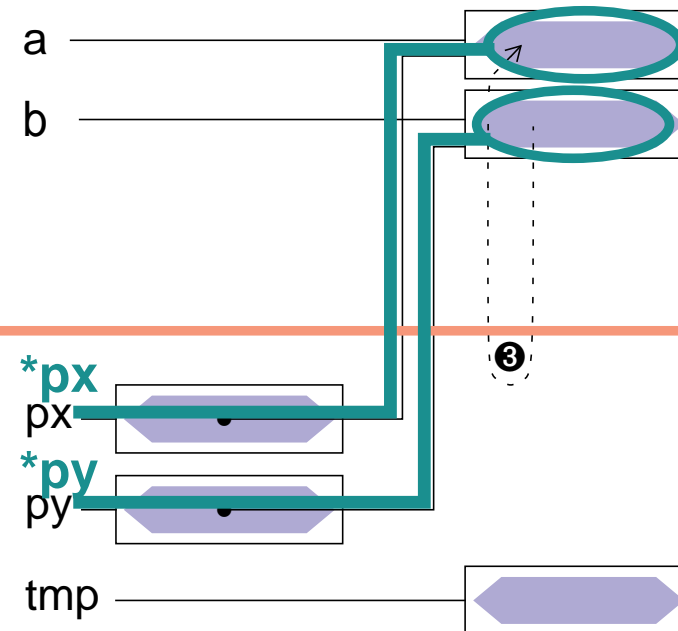
■ Beispiel:

```
void swap (double *, double *);
int main(void) {
double a, b;
...
swap(&a, &b); ❶
}
```

```
void swap (double *px, double *py)
{
double tmp;

tmp = *px;
*px = *py; ❷
*py = tmp;

}
```



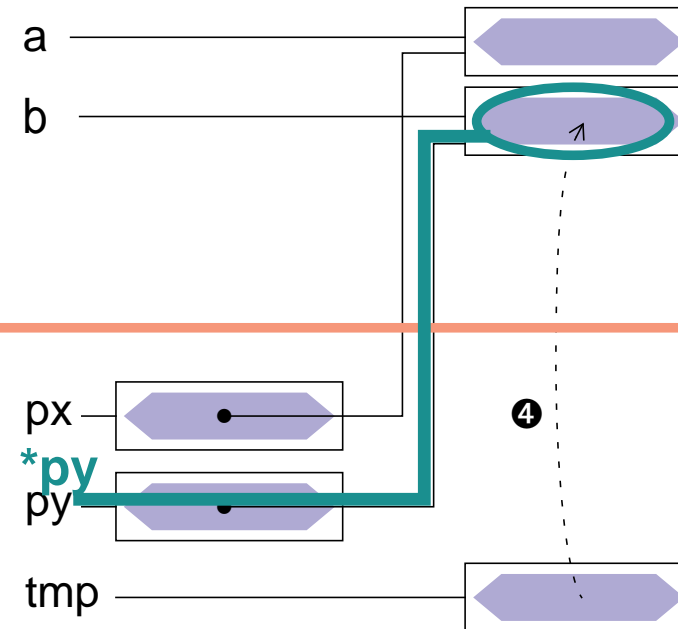
1 Zeiger als Funktionsargumente

■ Beispiel:

```
void swap (double *, double *);
int main(void) {
double a, b;
...
swap(&a, &b); ①
}
```

```
void swap (double *px, double *py)
{
double tmp;

tmp = *px; ②
*px = *py; ③
*py = tmp; ④
}
```



U3-2 generische swap-Funktion

- Funktion soll Zeiger auf beliebigen Datentyp übergeben bekommen

? welchen Typ gibt man dem Parameter

- ▶ Typ `(void *)` = Zeiger auf "irgendetwas"

- ▶ Schnittstelle der Funktion

```
void swap_generic(void *px, void *py, size_t s)
```

? wie benutzt man so einen Zeiger

- ▶ er kann nicht direkt genutzt werden, weil für das Ergebnis von `*px` und `*py` der Typ unbekannt ist

- => Programm kann nicht damit umgehen

- ▶ Lösung: void-Zeiger in einen anderen Zeiger verwandeln

- => cast-Operator

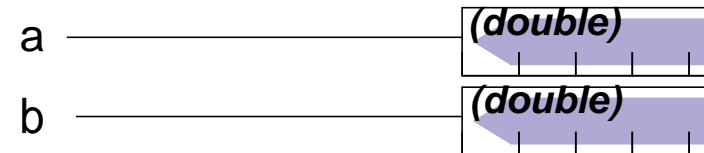
- ▶ Beispiel: `char *pa = (char *)px;`

- ▶ über `*pa` kann nun auf das erste Byte des Speicherbereichs, auf den `px` zeigt, zugegriffen werden

1 generische Zeiger als Funktionsargumente

■ Beispiel:

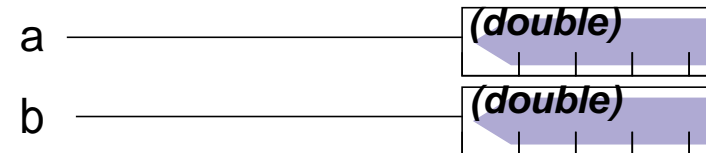
```
main(void) {  
    double a, b;  
    void swap_generic(void *, void *, size_t);  
    ...  
    swap_generic(&a, &b, sizeof(double));  
}
```



1 generische Zeiger als Funktionsargumente

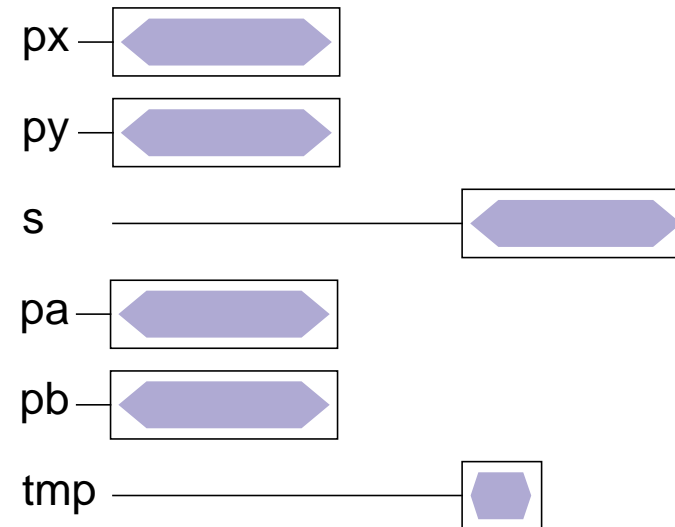
■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```



```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
}
```



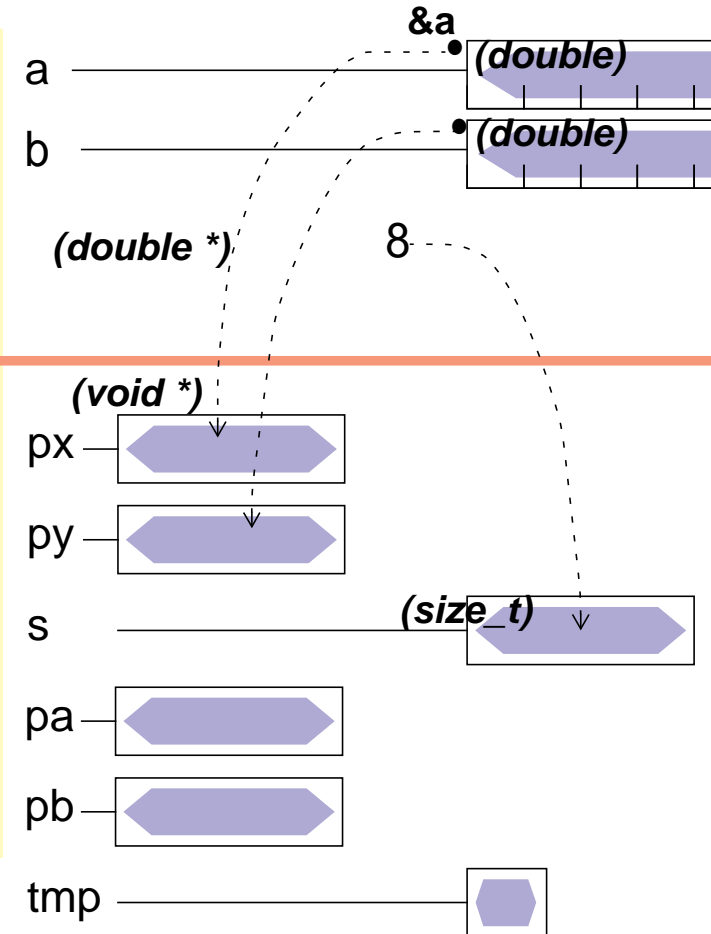
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double)); }

void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
}
```



double-Zeiger &a, &b werden an void-Zeiger px, py übergeben!!!

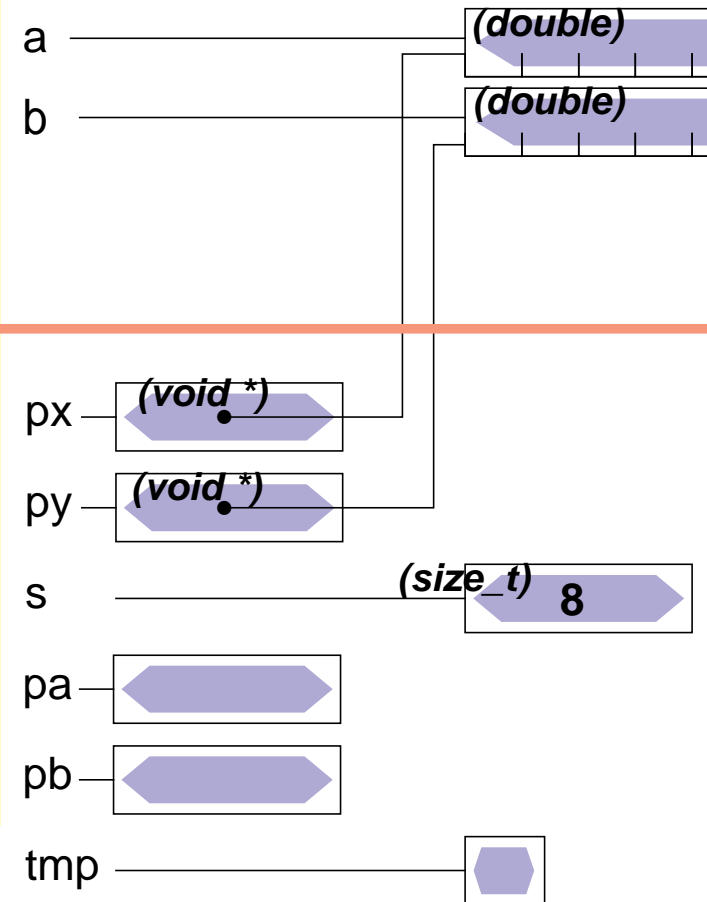
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
}
```



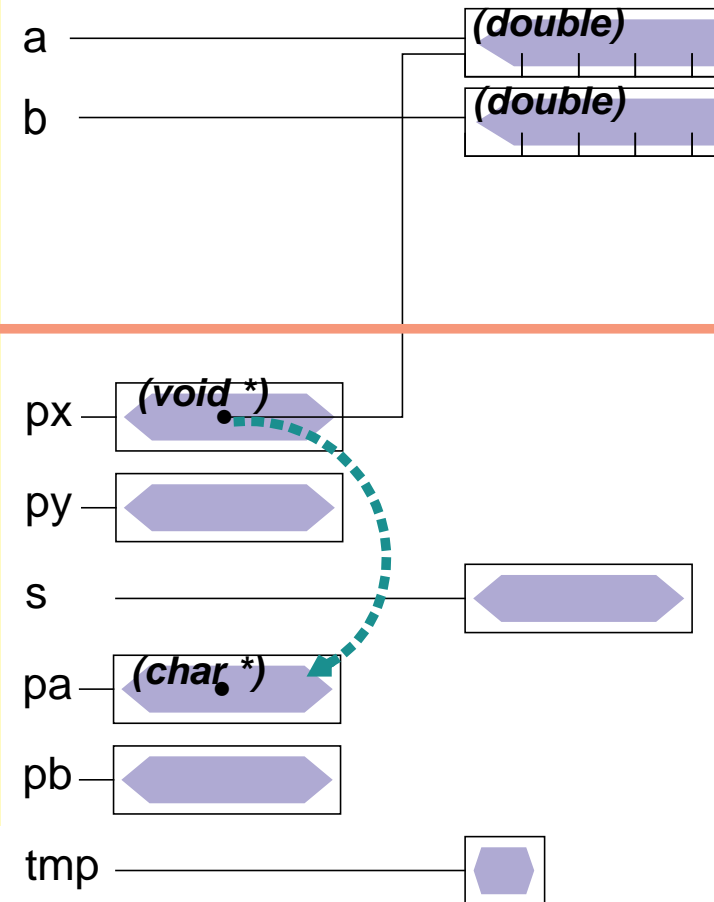
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
}
```



void-Zeiger px wird in char-Zeiger verwandelt und pa zugewiesen!

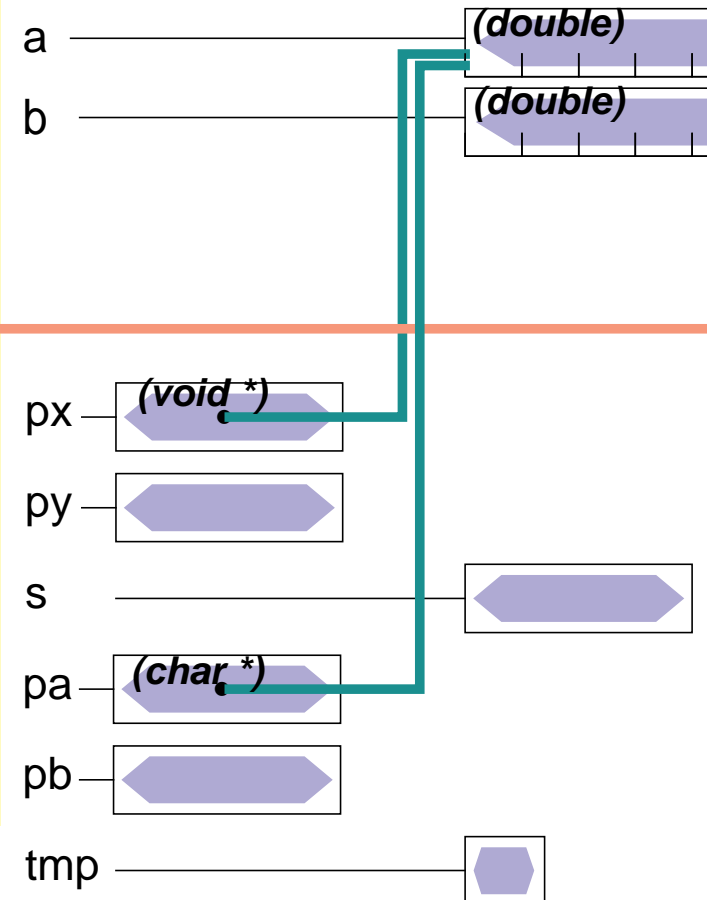
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
}
```



zwei Zeiger mit unterschiedlichem Typ zeigen jetzt auf gleiche Speicherstelle (Variable a)!

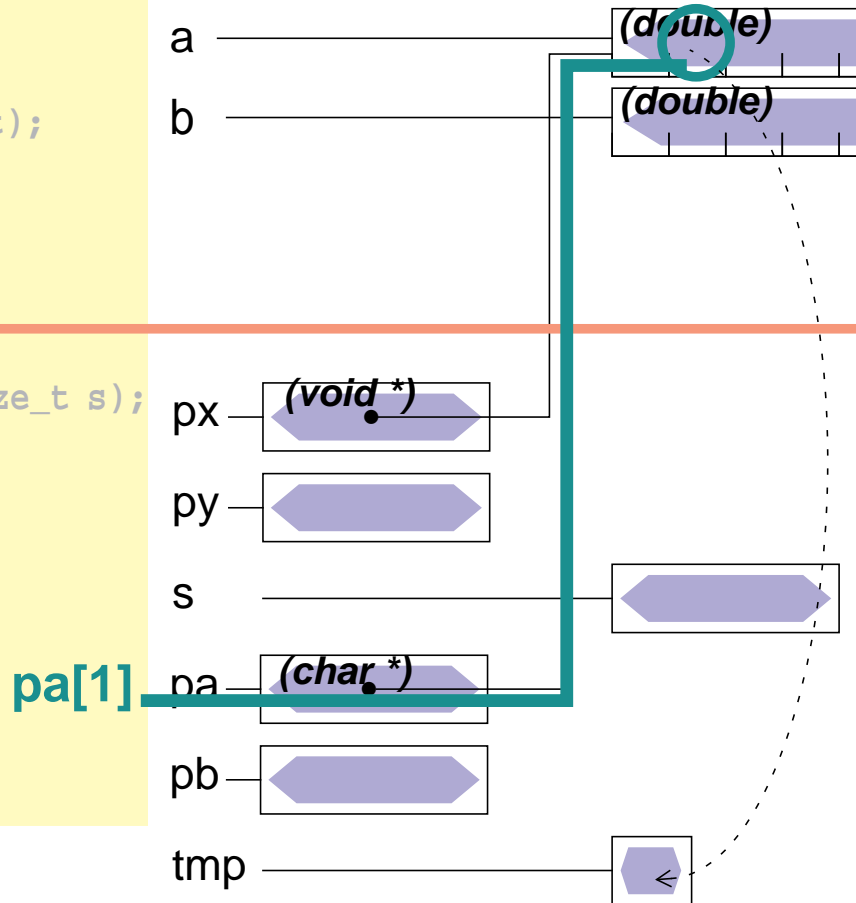
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
    tmp = pa[1];
```



pa wird als char-Array betrachtet!

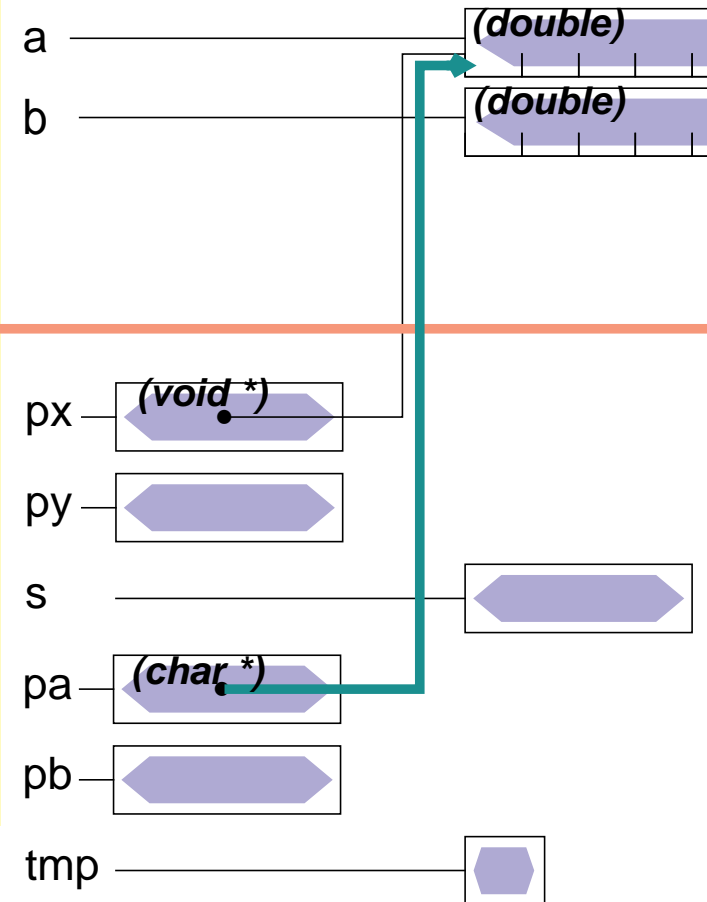
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
}
```



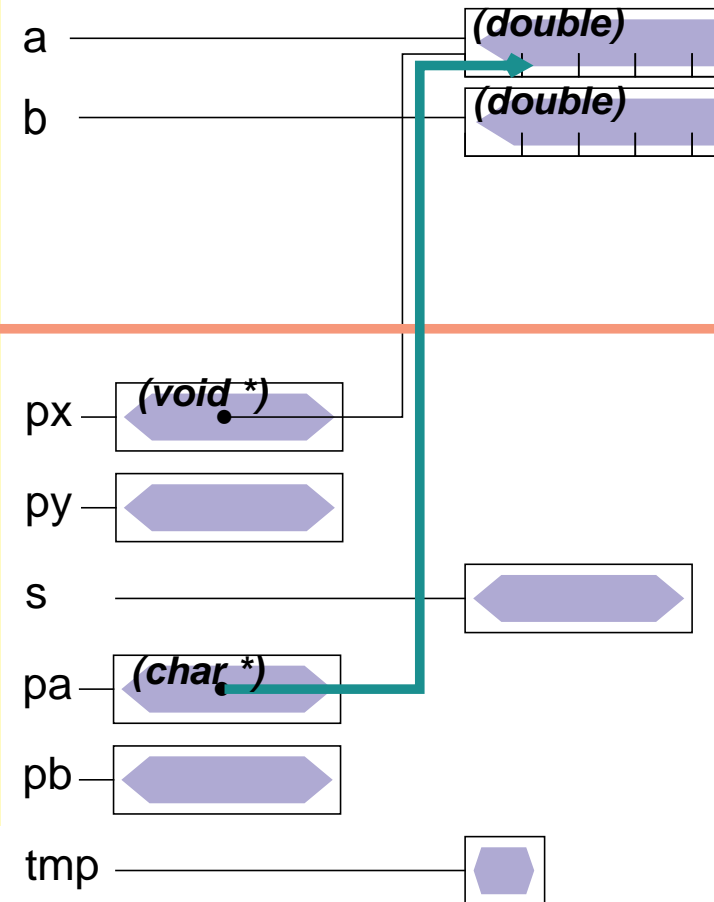
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
    pa++;
}
```



pa wird als char-Zeiger betrachtet und inkrementiert - zeigt jetzt auf das zweite Byte von a!

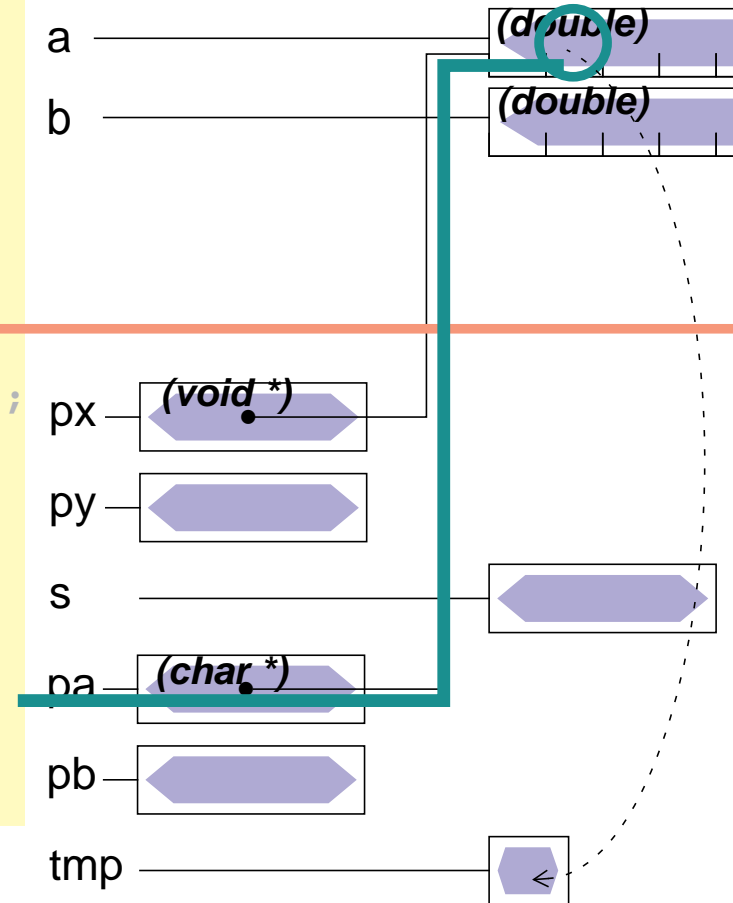
1 generische Zeiger als Funktionsargumente

■ Beispiel:

```
main(void) {
double a, b;
void swap_generic(void *, void *, size_t);
...
swap_generic(&a, &b, sizeof(double));
}
```

```
void swap_generic(void *px, void *py, size_t s);
{
    char *pa, *pb, tmp;

    pa = (char *)px;
    ...
    pa++;
    tmp = *pa;
```



zweites Byte der Variablen a wird in tmp zwischengespeichert!

