Bendar í C

Tölvunarfræði 2, vor 2012

Hallgrímur H. Gunnarsson

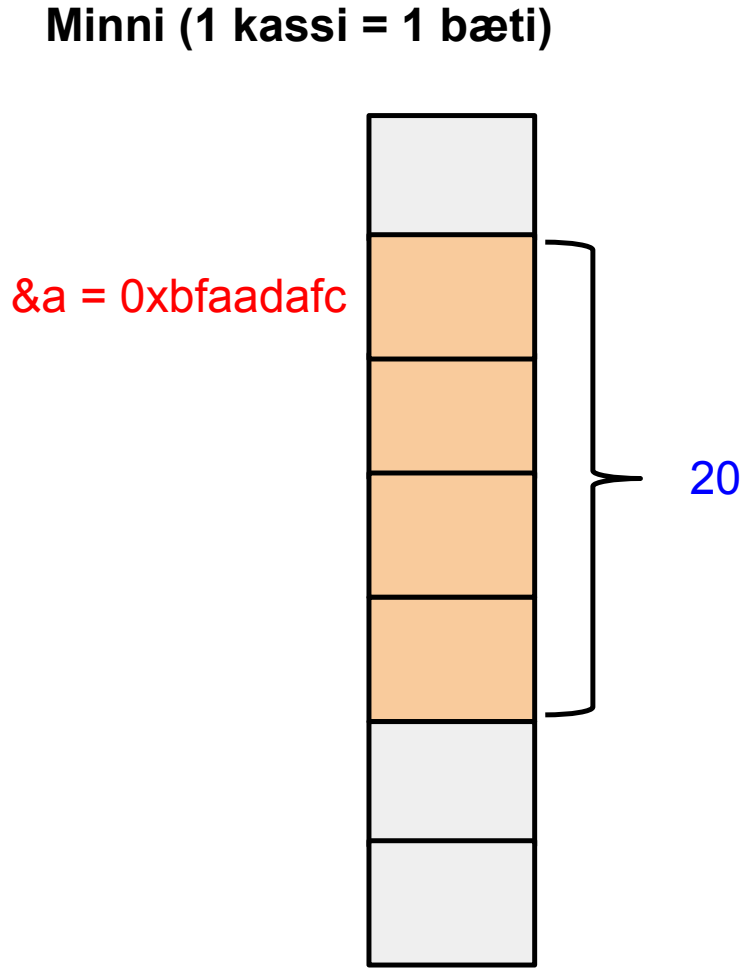
Háskóli Íslands

2012-01-13

Svæði breytu í minni

```c
#include <stdio.h>

int main(void)
{
    int a = 20;

    printf("%d\n", a);
    printf("%d\n", sizeof(a));
    printf("%p\n", &a);
}
```

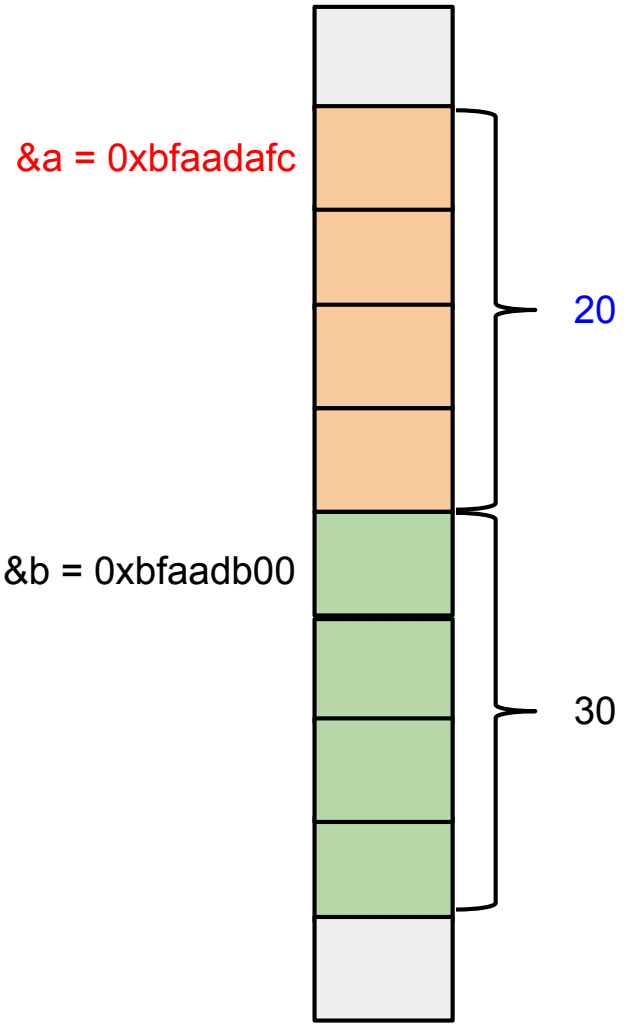
```
hhg@hhg:~/t2$ ./t
20
4
0xbfaadafc
hhg@hhg:~/t2$
```



Tvær breytur

```
int main(void)
{
    int a = 20;
    int b = 30;

    printf(" a = %d\n", a);
    printf("&a = %p\n", &a);
    printf(" b = %d\n", b);
    printf("&b = %p\n", &b);
}
```

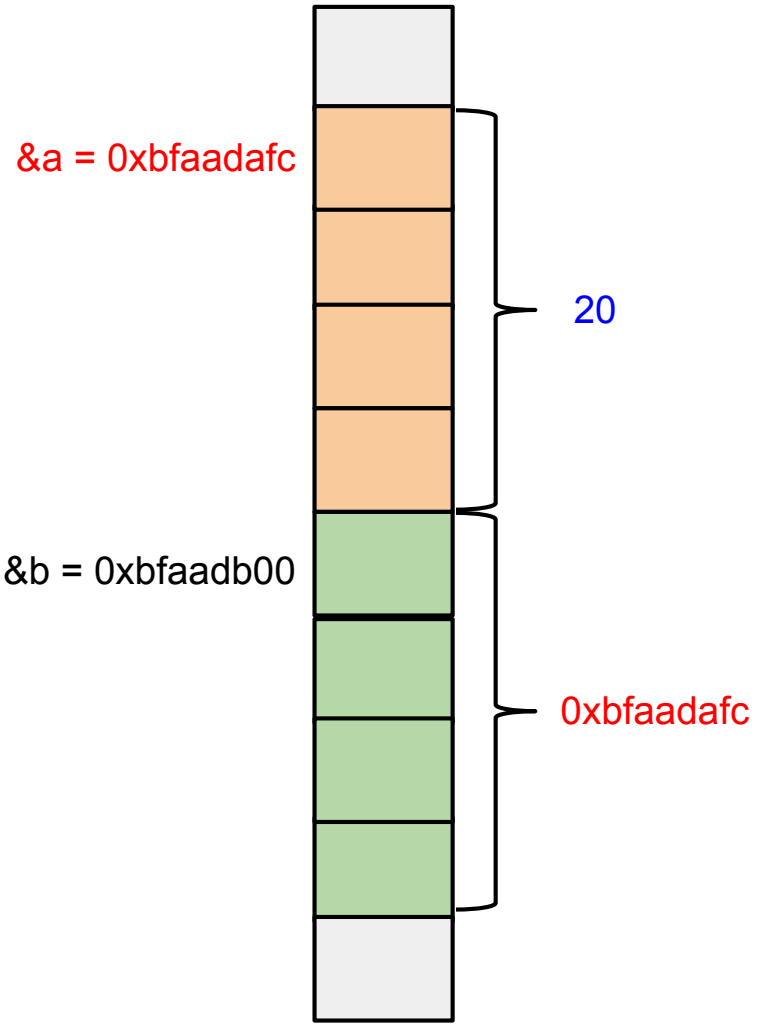
```
hhg@hhg:~/t2$ ./t2
 a = 20
&a = 0xbfaadafc
 b = 30
&b = 0xbfaadb00
hhg@hhg:~/t2$
```



Ein breyta sem bendir á aðra

```
int main(void)
{
    int a = 20;
    int *b = &a;

    printf(" a = %d\n", a);
    printf("&a = %p\n", &a);
    printf(" b = %p\n", b);
    printf("&b = %p\n", &b);
}
```

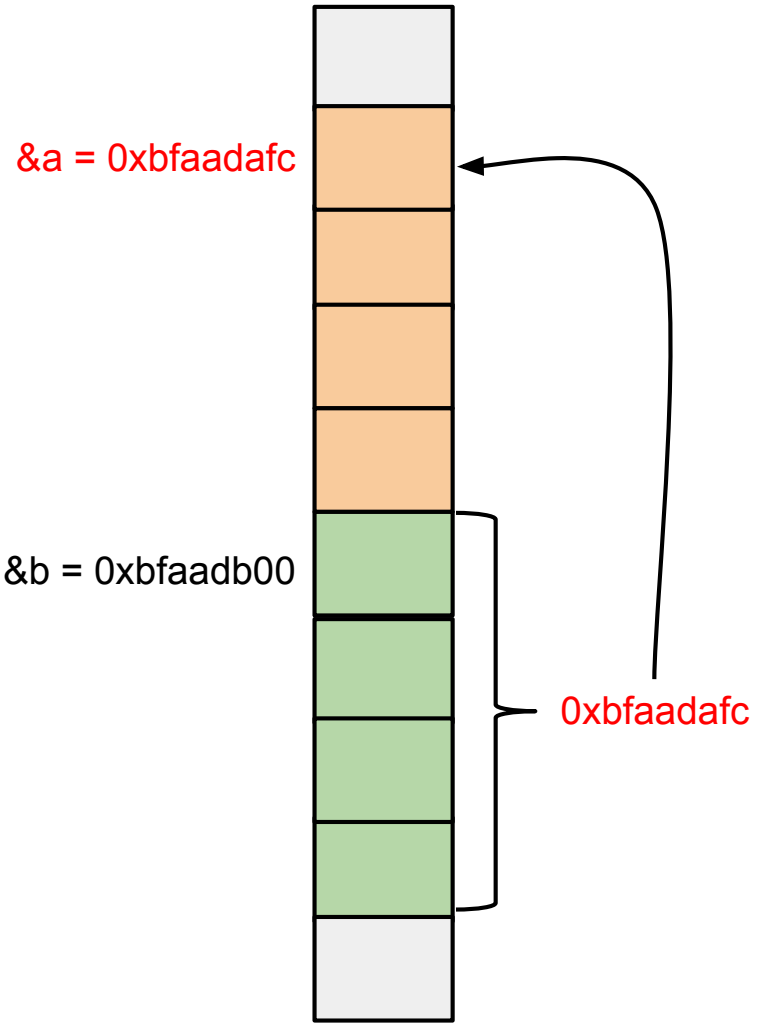
```
hhg@hhg:~/t2$ ./t3
 a = 20
&a = 0xbfaadafc
 b = 0xbfaadafc
&b = 0xbfaadb00
hhg@hhg:~/t2$
```



Ein breyta sem bendir á aðra

```
int a = 20;
int *b = &a;

printf(" a = %d\n", a);
printf("&a = %p\n", &a);
printf(" b = %p\n", b);
printf("&b = %p\n", &b);
```

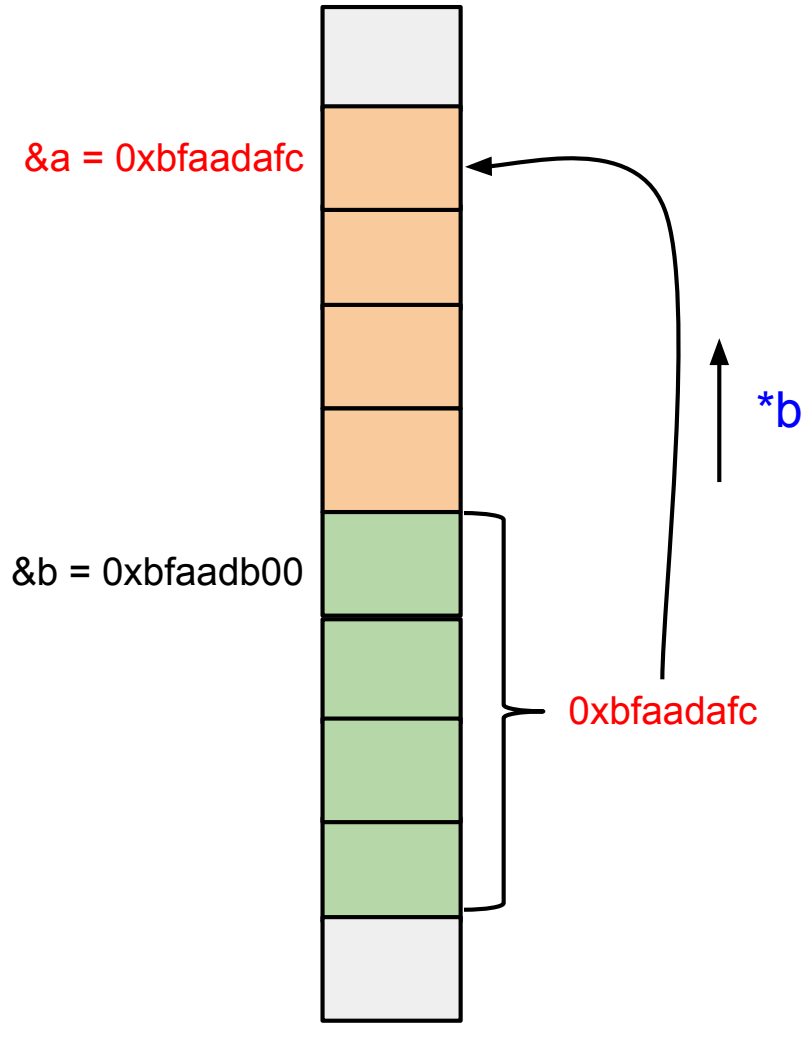
```
hhg@hhg:~/t2$ ./t3
 a = 20
&a = 0xbfaadafc
 b = 0xbfaadafc
&b = 0xbfaadb00
hhg@hhg:~/t2$
```



Sækja gildi með bendi (eltum bendi)

```
int a = 20;
int *b = &a;

printf(" a = %d\n", a);
printf("&a = %p\n", &a);
printf(" b = %p\n", b);
printf("&b = %p\n", &b);
printf("*b = %d\n", *b);
```

```
hhg@hhg:~/t2$ ./t3
 a = 20
&a = 0xbfaadafc
 b = 0xbfaadafc
&b = 0xbfaadb00
*b = 20
hhg@hhg:~/t2$
```



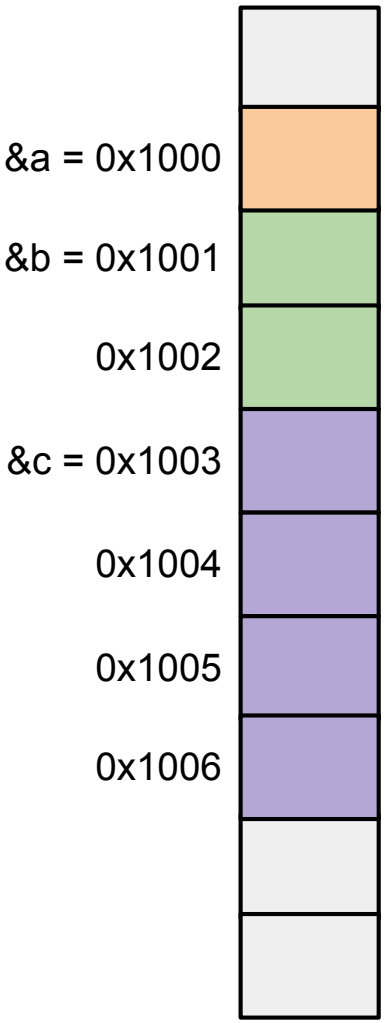
Stærð benda

- Bendir er breyta sem geymir minnisvistfang ("vísar á minni")
- Stærð bendis ræðst af stærð minnisvistfanga á tölvunni
  - 4 bæti á 32-bita tölvum
  - 8 bæti á 64-bita tölvum
- Bendir hefur tag (t.d. int *, char *, long *) en stærðin er alltaf sú sama, enda inniheldur bendir bara minnisvistfang

```
int *a;
char *b;
long *c;

printf("a = %d\n", sizeof(a));
printf("b = %d\n", sizeof(b));
printf("c = %d\n", sizeof(c));
```

```
hhg@hhg:~/t2$ ./t5
a = 4
b = 4
c = 4
hhg@hhg:~/t2$
```

Tag benda



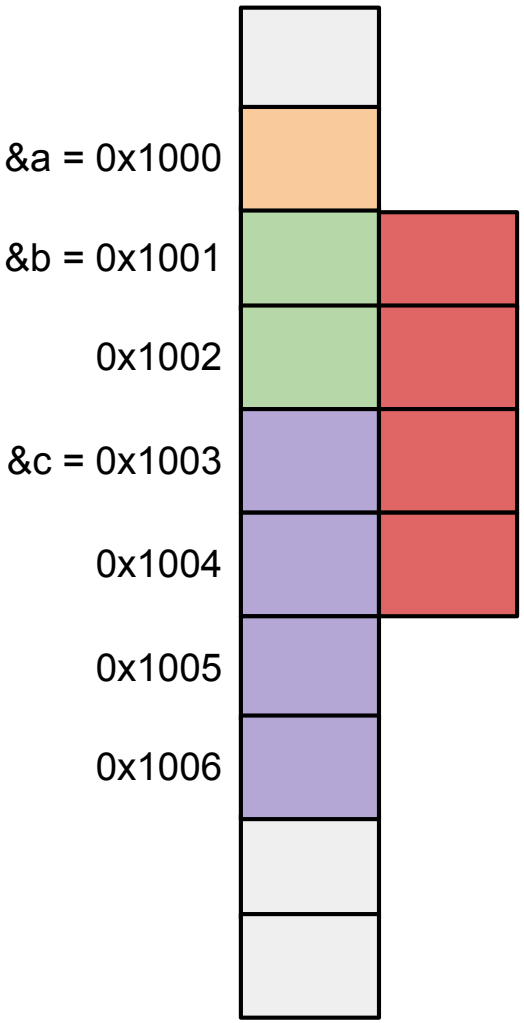
- Bendir hefur tag:

  ```
  int *p;
  ```

  er lesið sem: *p er bendir á int*, þ.e. p vísar á minnisvistfang sem inniheldur int
- Tagið er notað til að afmarka svæðið sem er bent á.

  ```
  char a = 'x';
  short b = 10;
  int c = 20;
  int *p = &c;
  ```
- p vísar á &c, sem er upphafið á minnissvæði c. Tagið (int) segir okkur hvar svæðið endar.
- int tekur 4 bæti þ.a. þegar við gerum *p þá lesum við 0x1003 upp í 0x1006

Brjótum reglurnar



- Höfum eftirfarandi breytur:

```
char a = 'x';
short b = 10;
int c = 20;
```

- Hvað gerist ef við látum int *p vísa á &b, sem er short?

```
int *p = &b;
```

- Við fáum warning frá C þýðendanum, en hann leyfir okkur samt að gera þetta
- Þegar við gerum *p þá eru rauðu reitirnir lesnir og túlkaðir sem "int" gildi
- Fáum hluta úr b og hluta úr c, túlkað saman sem int

Orðaforði

Vistfang í minni (address of):

- Einundaraðgerðin & (address-of operator) skilar minnisvistfangi breytu

  ```
  int *p = &a;
  ```

  lætur `p` benda á minnisvistfang `a`
- `&a` er lesið sem: vistfang a (address of a)

Sækja gildi (dereference):

- Einundaraðgerðin $*$ (dereference operator) eltir bendir og skilar því sem hann bendir á
- `*p` er lesið sem: gildið sem p bendir á (the value pointed at by p)

Gildisveiting með bendi

```c
int a = 10;
int *p = &a;

printf("a = %d\n", a);
printf("*p = %d\n", *p);
a = 20;
printf("a = %d\n", a);
printf("*p = %d\n", *p);
```

```
hhg@hhg:~/t2$ ./ab
a = 10
*p = 10
a = 20
*p = 20
hhg@hhg:~/t2$
```

```c
int a = 10;
int *p = &a;

printf("a = %d\n", a);
printf("*p = %d\n", *p);
*p = 20;
printf("a = %d\n", a);
printf("*p = %d\n", *p);
```

```
hhg@hhg:~/t2$ ./ab
a = 10
*p = 10
a = 20
*p = 20
hhg@hhg:~/t2$
```

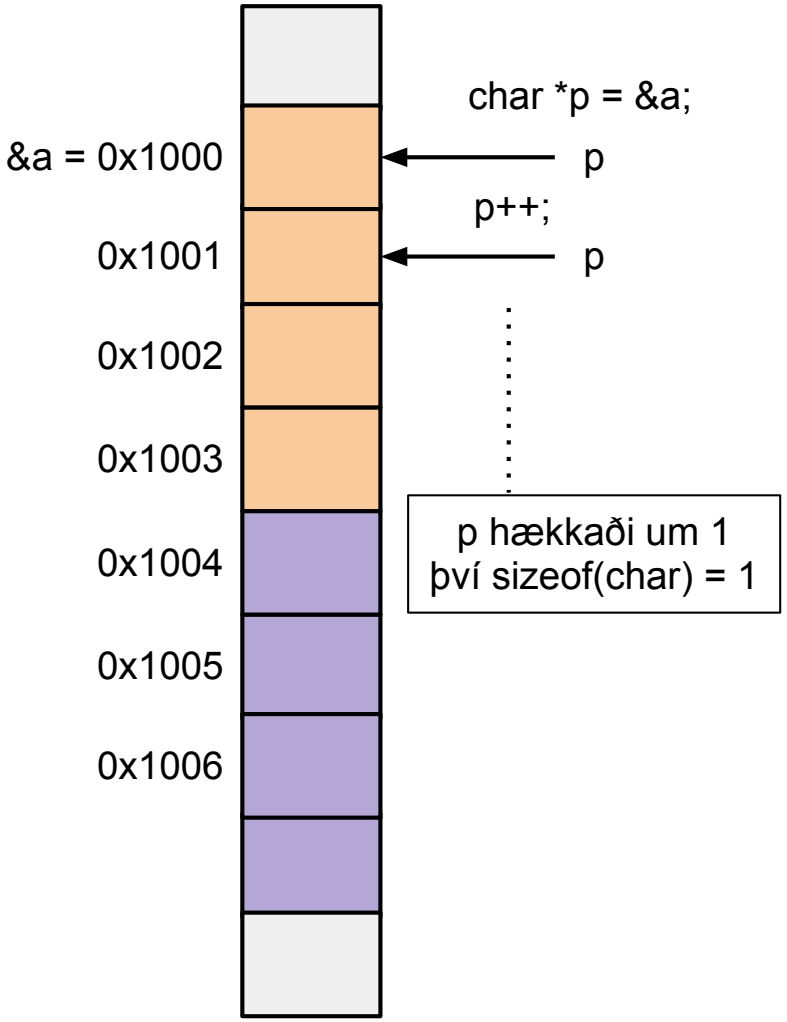
Dæmi um swap fall

```
void swap(int *a, int *b)
{
  int c = *a;
  *a = *b;
  *b = c;
}

int main(void)
{
  int a = 10;
  int b = 20;

  printf("a=%d b=%d\n", a, b);
  swap(&a, &b);
  printf("a=%d b=%d\n", a, b);
}
```

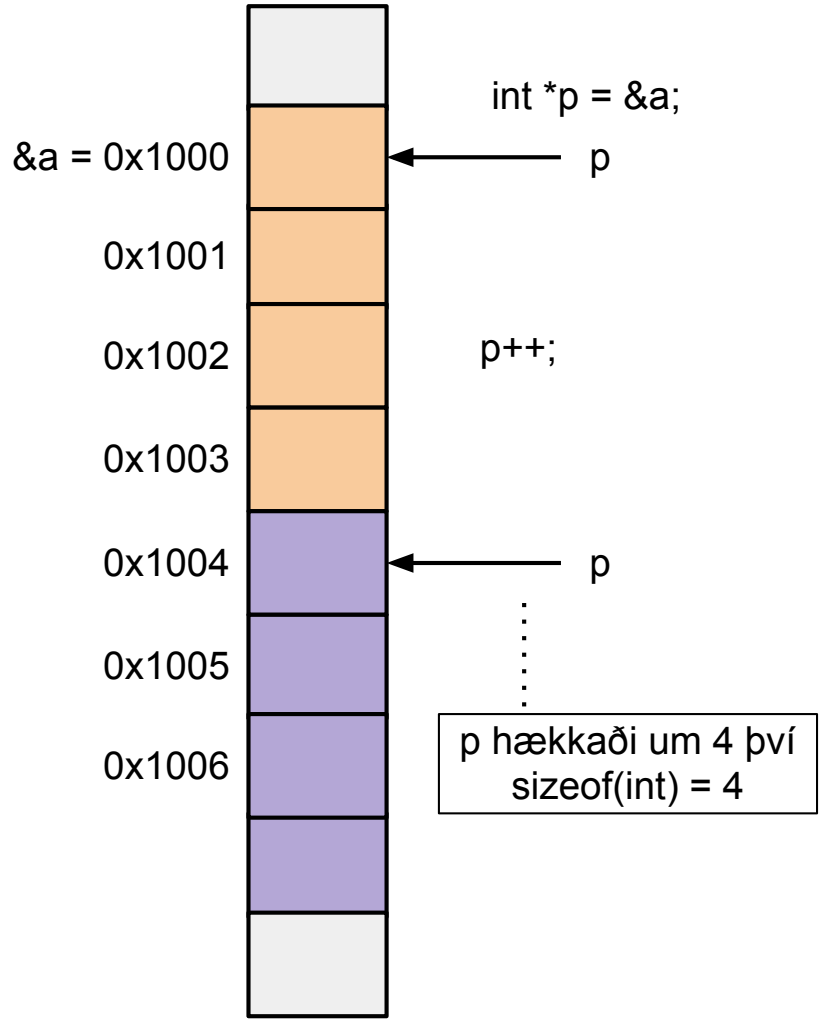
```
hhg@hhg:~/t2$ ./sw
a=10 b=20
a=20 b=10
hhg@hhg:~/t2$
```

Bendareikningur (e. pointer arithmetic)

char bendir:



int bendir:



Fylki og bendar

```c
#include <stdio.h>

int main(void)
{
  int a[10];
  int i;

  for (i = 0; i < 10; i++)
    a[i] = i;

  int *p = &a[0];
  printf("%d\n", *p);
  p++;
  printf("%d\n", *p);
}
```

```
hhg@hhg:~/t2$ ./t10
0
1
hhg@hhg:~/t2$
```

- Gildið á *a*, þar sem *a* er fylki, er í raun vistfangið á fyrsta stakinu í fylkinu
- Gildisveitingin

  ```c
  int *p = &a[0];
  ```

  er því jafngild

  ```c
  int *p = a;
  ```

Strengir

Munið: strengir eru bara fylki af bókstöfum!

```c
int main(void)
{
  char a[4] = "abc";
}
```

```c
int main(void)
{
  char a[] = "abc";
}
```

```c
int main(void)
{
  char a[4];

  a[0] = 'a';
  a[1] = 'b';
  a[2] = 'c';
  a[3] = 0;
}
```

Göngum yfir streng með fylkisvísun

```c
#include <stdio.h>

int main(void)
{
  char a[] = "abc";
  int n = strlen(a);
  int i;

  for (i = 0; i < n; i++)
  {
    printf("%c\n", a[i]);
  }
}
```

```
hhg@hhg:~/t2$ ./t11
a
b
c
hhg@hhg:~/t2$
```

Göngum yfir streng með bendi

```c
#include <stdio.h>

int main(void)
{
  char a[] = "abc";
  char *p = a;

  while (*p)
  {
    printf("%c\n", *p);
    p++;
  }
}
```

```
hhg@hhg:~/t2$ ./t12
a
b
c
hhg@hhg:~/t2$
```