A Brief Review of C
(and Beards)

Benjamin Brewster

A Brief History

- Developed by Dennis Ritchie (1941-2011) between 1969 and 1973 at Bell Labs

- C is a successor to B; however, B's inability to take advantage of the PDP-11's advanced features (to which computer Ritchie and Ken Thompson were busily porting UNIX) caused Ritchie to develop C

- UNIX was then re-written in C in 1972, which had been in development at the same time

UNIX Beard Comparison – Round 1

Dennis Ritchie – restrained, non-ironic

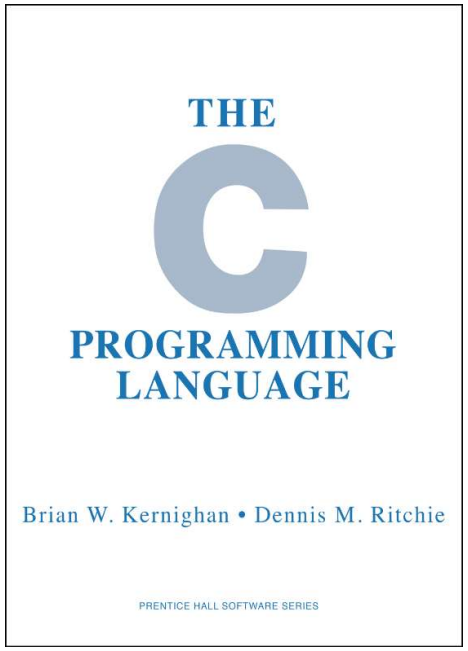
Richard Stallman – enough said

C is A High-Level Language

- As opposed to a low level language, like assembly
- The original version of C (C89) has 32 reserved keywords, and 50+ operators and syntax characters
- C syntax widely influences programming language syntax development today

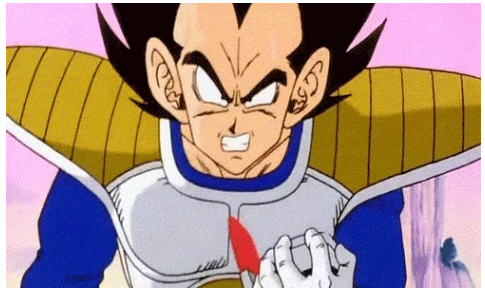
HELLO FREAKING WORLD

```
#include <stdio.h>

int main()
{
  printf("Hello world\n");
  return 0;
}
```



The first C book, written by Ritchie and Brian Kernighan, contains the first usage of a Hello World program put in book form

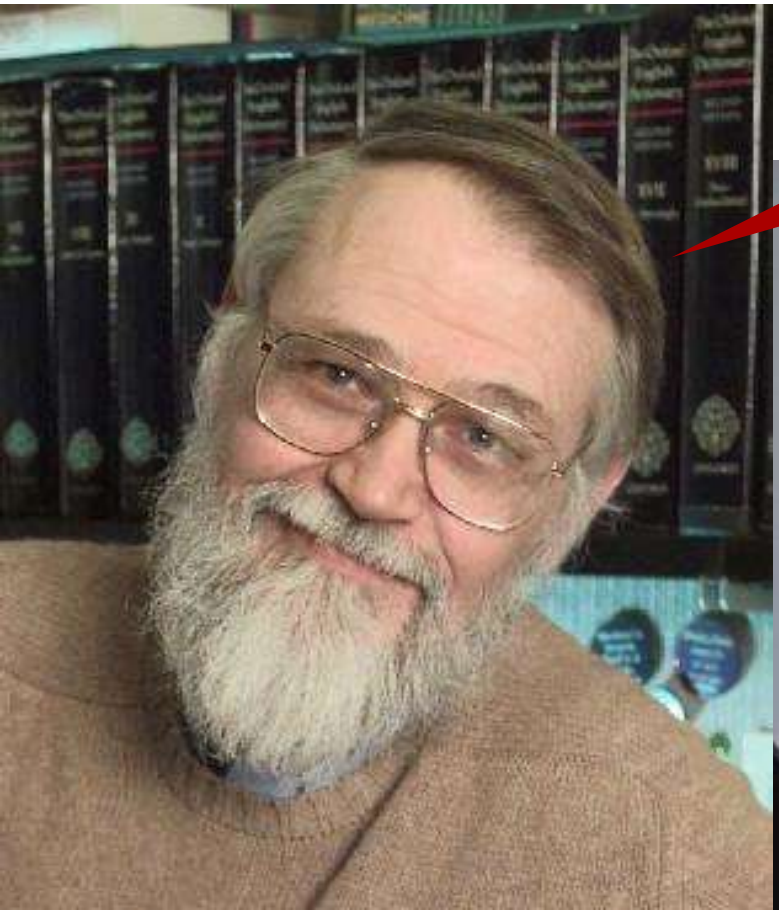
PART DEUX



```
#include <stdio.h>

int main()
{
    char* oblig = "Hello World";
    float itsOver = 9000.0f;
    printf("%s\n", oblig);
    printf("IT\'S OVER %.2f!\n", itsOver);

    return 0;
}
```

```
$ hw2
Hello World
IT'S OVER 9000.00!
```

UNIX Beard Comparison – Round 2



Brian Kernighan – shaped yet voluminous

Richard Stallman – enough said

Common String Shenanigans - Comparing

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

void main()
{
        char* boring = "boring";
        char* weirdDadSaying = "Eat more beef, kick less cats\n";
        int length;

        length = strlen(weirdDadSaying);
        printf("Length of entered string is = %d\n", length);

        if (strcmp(boring, weirdDadSaying) == 0)
                printf("Entered strings are equal.\n");
        else
                printf("Entered strings are not equal.\n");
}
```

```
$ stringshenanigans
Length of entered string is = 30
Entered strings are not equal.
```

Why Only Two Arguments? That's Weird Design

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

void main()
{
   char a[1000], b[1000];

   printf("Enter the first string\n");
   gets(a);

   printf("Enter the second string\n");
   gets(b);

   strcat(a, b);

   printf("String obtained on concatenation is %s\n", a);
 }
```

Function signature:
`char* gets(char *str)`

`strcat()` dumps the results into a and returns the same

Why Only Two Arguments? That's Weird Design

```
$ gcc -o getsstrcat getsstrcat.c
/tmp/ccJ1Kg0x.o: In function `main':
getsstrcat.c:(.text+0x20): warning: the `gets' function is dangerous and should not be used.
$ getsstrcat
Enter the first string
mystring!
Enter the second string
so col!!@
String obtained on concatenation is mystring!so col!!@
```

```
  printf("Enter the second string\n");
  gets(b);

  strcat(a, b);

  printf("String obtained on concatenation is %s\n", a);
}
```

`strcat()` dumps the results into a and returns the same

Substrings - Not Built-In!

```
#include <stdio.h>

void main()
{
   char string[1000], sub[1000];
   int position, length, c = 0;

   printf("Input a string\n");
   gets(string);

   printf("Enter the position of first char, a space, and length of substring\n");
   scanf("%d%d", &position, &length);

   while (c < length) {
      sub[c] = string[position + c - 1];
      c++;
   }
   sub[c] = '\0';

   printf("Required substring is \"%s\"\n", sub);
}
```

Substrings - Not Built-In!

```
$ gcc -o substrings substrings.c
/tmp/ccdSGmo9.o: In function `main':
substrings.c:(.text+0x27): warning: the `gets' function is dangerous and should not be used.
$ substrings
Input a string
test string!
Enter the position of first char, a space, and length of substring
2 6
Required substring is "est st"
```

```
   printf("Enter the position of first char, a space, and length of substring\n");
   scanf("%d%d", &position, &length);

   while (c < length) {
      sub[c] = string[position + c - 1];
      c++;
   }
   sub[c] = '\0';

   printf("Required substring is \"%s\"\n", sub);
}
```

Array Stuff

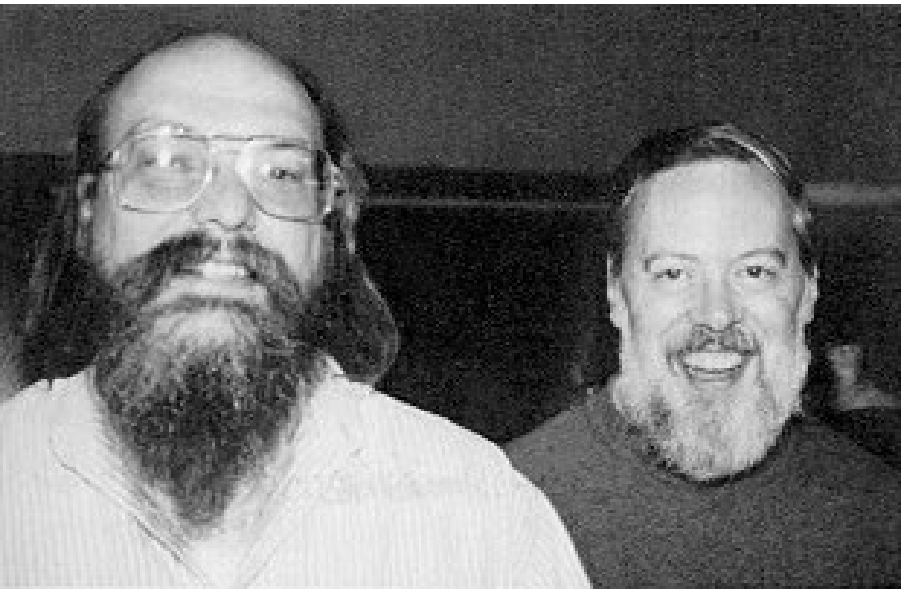
```
#include <stdio.h>

void main()
{
  int array[100], maximum, size, c, location = 1;

  printf("Enter the number of elements in array\n");
  scanf("%d", &size);

  printf("Enter %d integers\n", size);

  for (c = 0; c < size; c++)
    scanf("%d", &array[c]);

  maximum = array[0];

  for (c = 1; c < size; c++)
  {
    if (array[c] > maximum)
    {
       maximum  = array[c];
       location = c + 1;
    }
  }

  printf("Max element at location %d, value is %d.\n", location, maximum);
}
```

```
$ gcc -o arraystuff arraystuff.c
$ arraystuff
Enter the number of elements in array
5
Enter 5 integers
1 9 3 7 4
Max element at location 2, value is 9.
```

UNIX Beard Comparison – Round 3

Ken Thompson – Unrestrained, yet directed

Bonus Dennis! Does this guy know how to party!



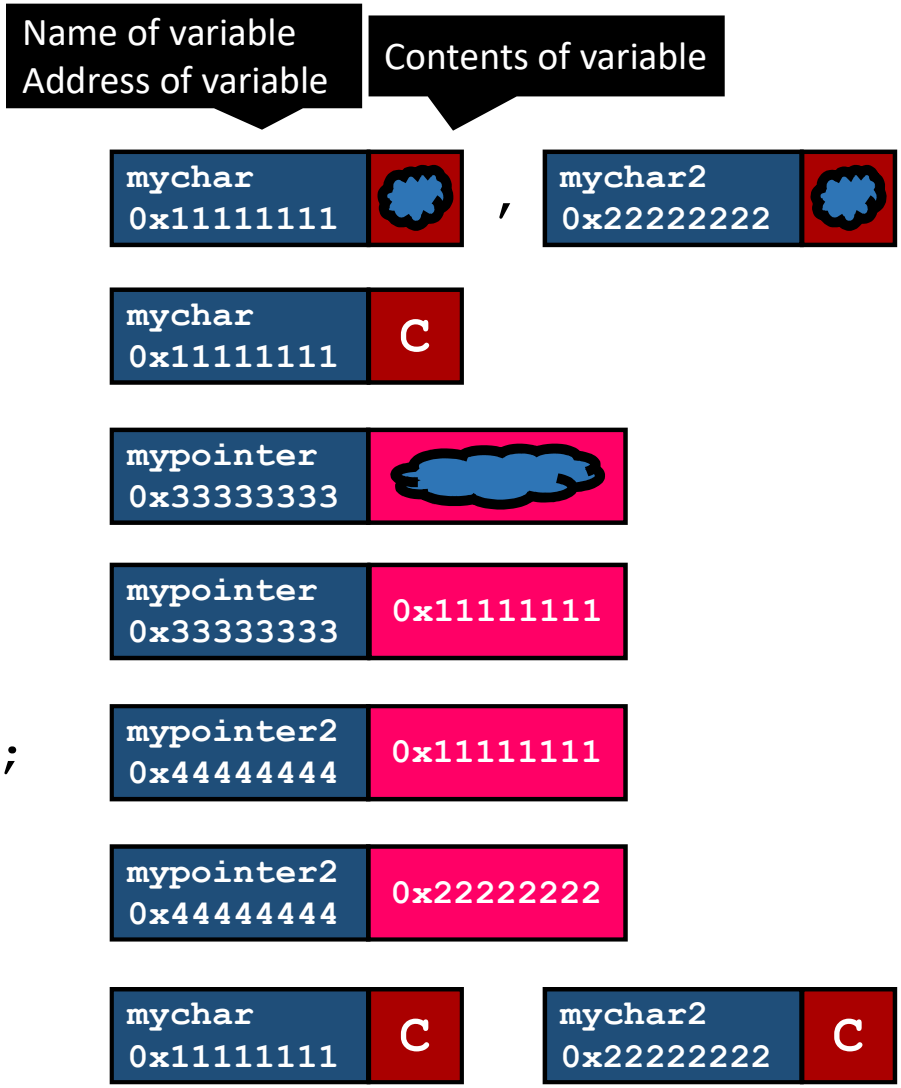
Gandalf the White

Richard Stallman

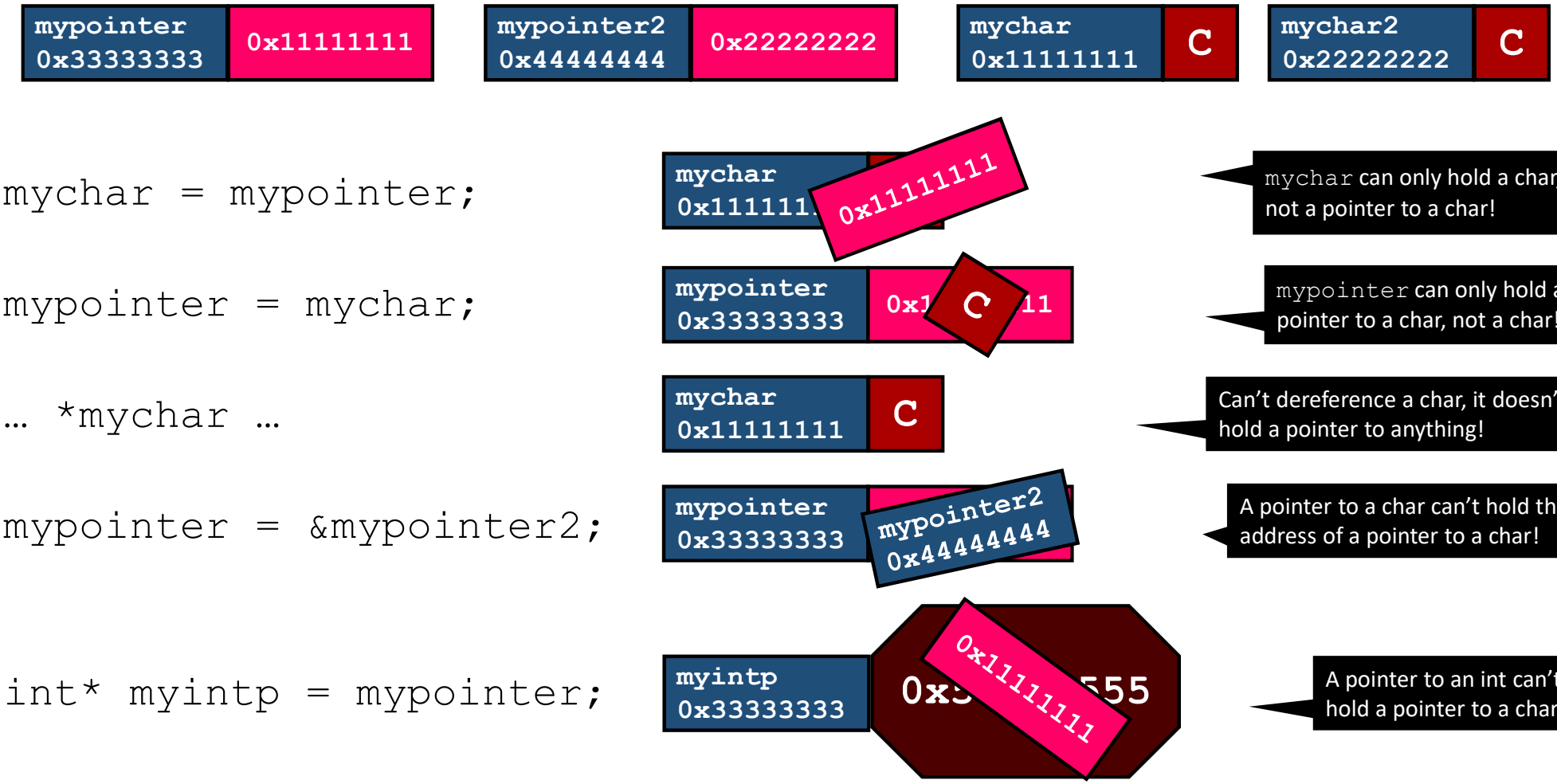
http://book.roomofthings.com/2012/05/15/enchanted_objects_IoT_for_humans

OH CRAP POINTERS

```
char mychar, mychar2;

mychar = 'C';

char* mypointer;

mypointer = &mychar;

char* mypointer2 = mypointer;

mypointer2 = &mychar2;

*mypointer2 = *mypointer;
```



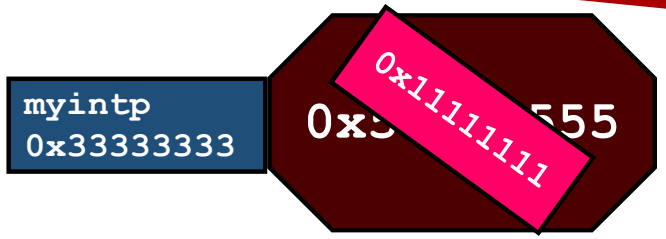
OH CRAP POINTERS - Illegal Commands

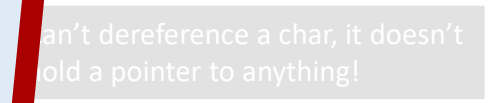


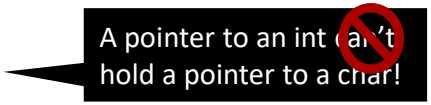
OH CRAP POINTERS - Illegal Commands


```
mychar = mypointer;

mypointer = mychar;

… *mychar …

mypointer = &mypointe

int* myintp = mypointer;
```

Except, C allows these four items, giving you a suitably dire **warning only** for each problem at compile time



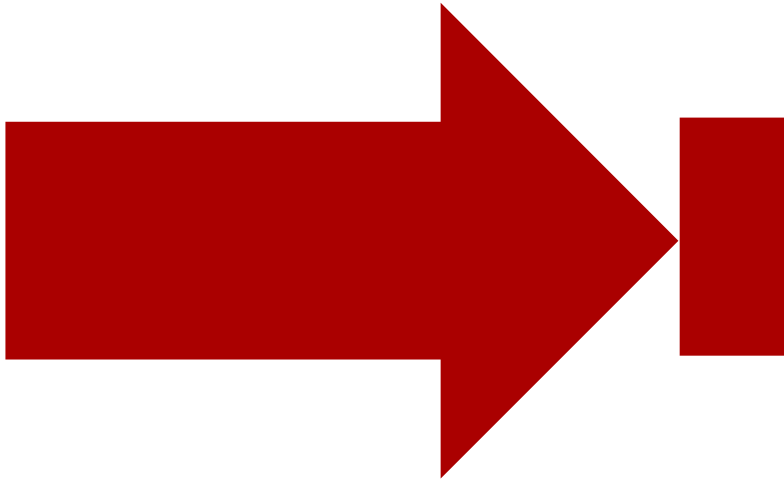




OH CRAP POINTERS

```
#include <stdio.h>

void CopyString(char* tgt, char* src)
{
        while (*src)
        {
                *tgt = *src;
                src++;
                tgt++;
        }

        *tgt = '\0';
}

void main()
{
        char target[1000];
        char* source = "COPY ME!";

        CopyString(target, source);
        printf("Target is: \'%s\'\n", target);
}
```

```
$ gcc -o ohcrappointers ohcrappointers.c
$ ohcrappointers
Target is: 'COPY ME!'
```



UNIX Beard Winner

Ed Gould
BSD Pioneer