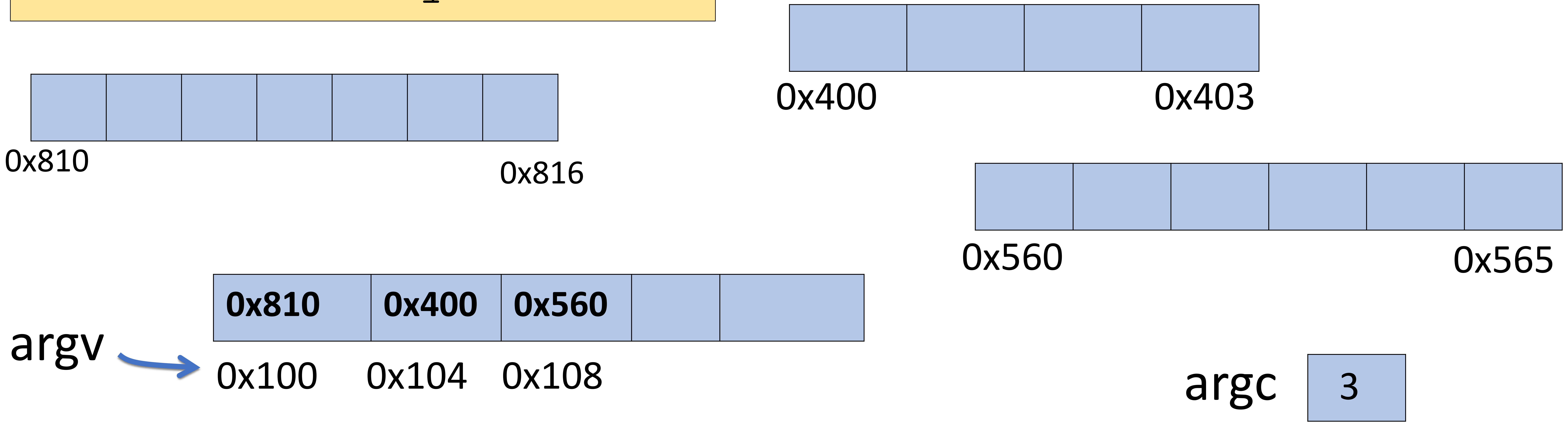
Argv is a Pointer to Pointers

```
int main (int argc, char **argv){
     …
}
```

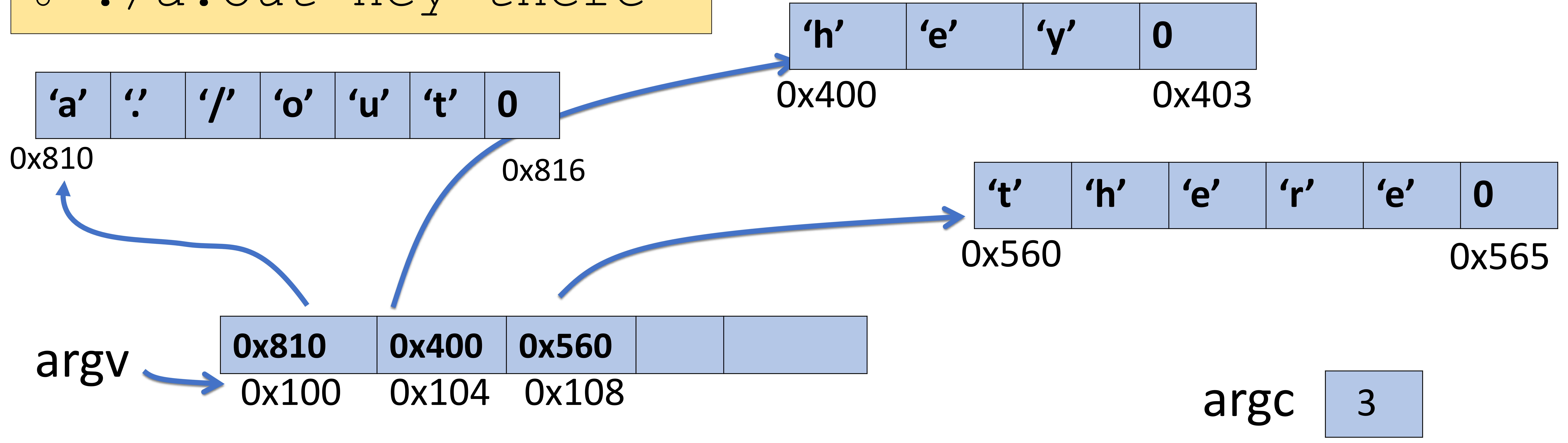
```
% ./a.out hey there
```



Argv is a Pointer to Pointers

```
int main (int argc, char **argv){
    …
}
```

```
% ./a.out hey there
```



Good news – array [] syntax works for pointers to arrays!!

Because char **argv is a pointer to an **array of char pointers**

- So argv[0] gives you a char *, which is a pointer to **an array of chars**
- Which means argv[0] gives you the first “string” in the array

Because argv[0] is a char * that is a pointer to **an array of chars**

- You can say argv[0][0] to get the first character in the first “string”

What is the output of this code?

```
int main (int argc, char **argv){
    printf(“%s”, argv[2]);
}
```

```
% ./a.out how are you?
```

A. `./a.out`
B. `how`
C. `are`
D. `you?`
E. `a`

What is the output of this code?

```
int main (int argc, char **argv){
    printf("%c", argv[1][2]);
}
```

```
% ./a.out how are you?
```

A. a

B. h

C. w

D. r

E. None of the above

What is the output of this code?

```
int main (int argc, char **argv){
    printf("%c", argv[1][3]);
}
```

```
% ./a.out how are you?
```

A. .

B.   ←Null char

C.   ←space

D. a

E. segfault

Let’s look at this in more detail

```
int main (int argc, char **argv){
    printf(“%c”, argv[1][3]);
}
```

```
% ./a.out how are you?
```

C Strings As Parameters

- When we pass a string as a parameter, it is passed as a **char ***
- C passes the location of the first character rather than a copy of the whole array

```
int doSomething(char *str) {
        ...
        str[0] = 'c’;          // modifies original string!
        printf("%s\n", str);   // prints cello
}

char myString[] = “Hello”;  // defines space and initializes
...
doSomething(myString);
```

Summary

- C is a valuable language that offers high performance
- Many programming constructs are similar between Java/C
    - Loops, if statements, etc.
- C programs have .h files in addition to .c files
- Arrays and Strings have important differences in C
    - Arrays can be allocated on the stack in C
    - Strings (just char[]) require null termination