Topic 10 – Strings/Word Wrapping

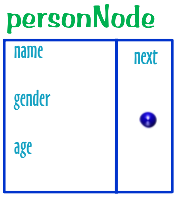
Pointers as parameters
Review on Strings
Word Wrapping

Chapter 12 in the online book

Pointers (Brief review)

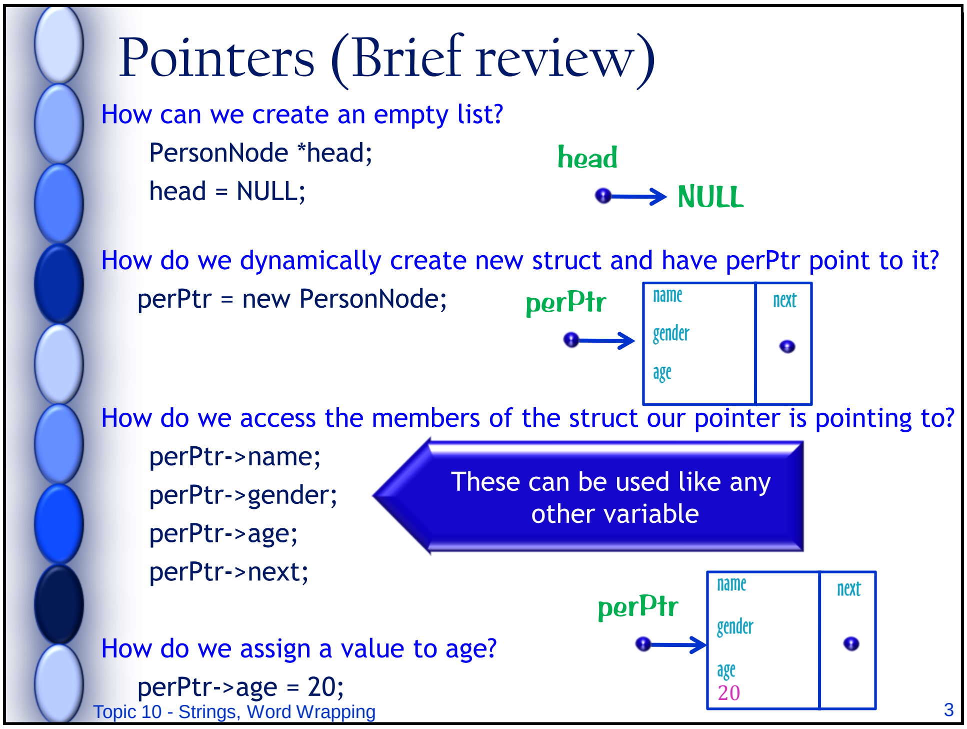
How can we declare a struct called PersonNode that contains a person’s name, gender, and age. This struct should be able to be used in a linked list

```
struct PersonNode
{
    string        name;
    char          gender;
    int           age;
    PersonNode *next;
};
```



How can we declare a pointer to our struct?

PersonNode *perPtr;



Pointers (Brief review)

How can we create an empty list?

```
PersonNode *head;
head = NULL;
```

head
NULL

How do we dynamically create new struct and have perPtr point to it?

```
perPtr = new PersonNode;
```

perPtr
name
gender
age
next

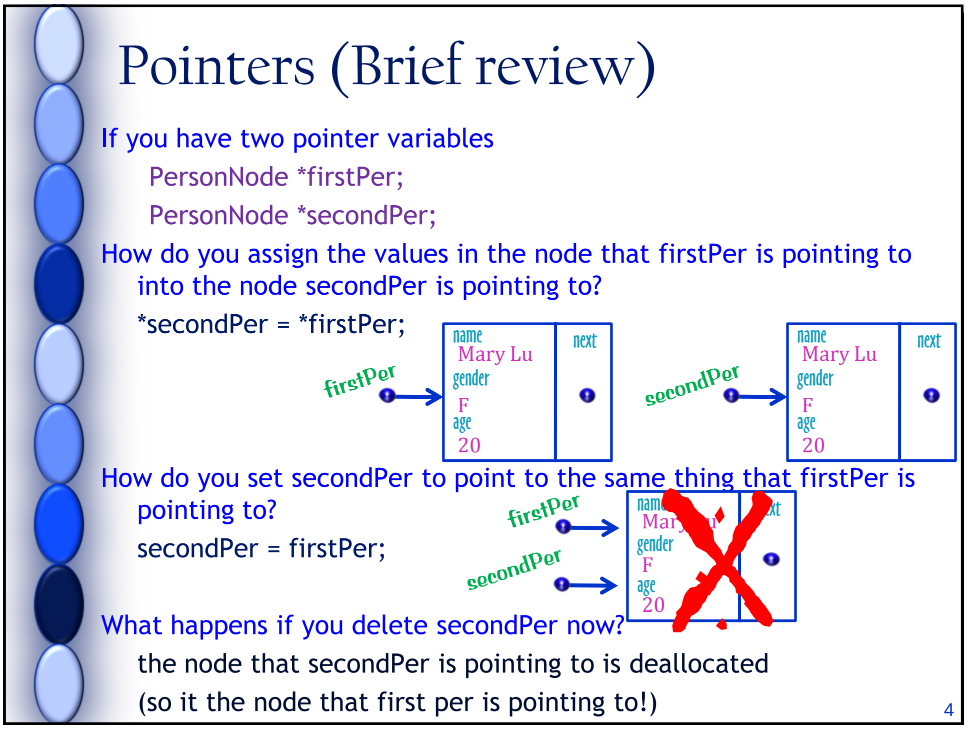
How do we access the members of the struct our pointer is pointing to?

```
perPtr->name;
perPtr->gender;
perPtr->age;
perPtr->next;
```

These can be used like any other variable

How do we assign a value to age?

```
perPtr->age = 20;
```



Pointers (Brief review)

If you have two pointer variables

```
PersonNode *firstPer;
PersonNode *secondPer;
```

How do you assign the values in the node that firstPer is pointing to into the node secondPer is pointing to?

```
*secondPer = *firstPer;
```

firstPer
name
Mary Lu
gender
F
age
20
next
secondPer
name
Mary Lu
gender
F
age
20
next

How do you set secondPer to point to the same thing that firstPer is pointing to?

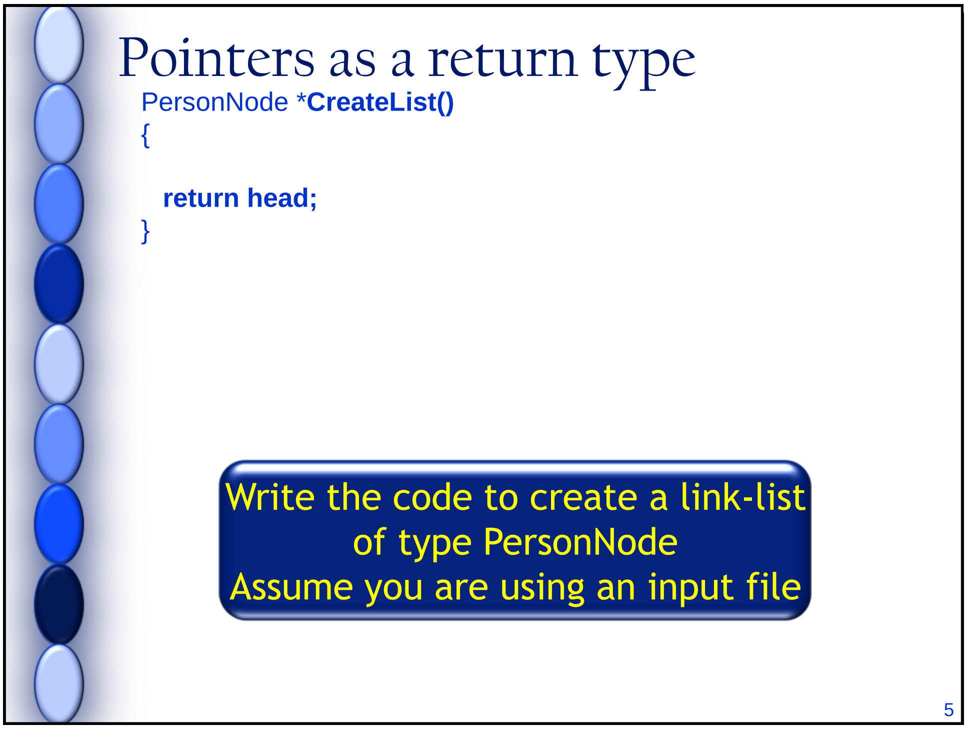
```
secondPer = firstPer;
```

firstPer
secondPer
name
Mary Lu
gender
F
age
20
next

What happens if you delete secondPer now?

the node that secondPer is pointing to is deallocated

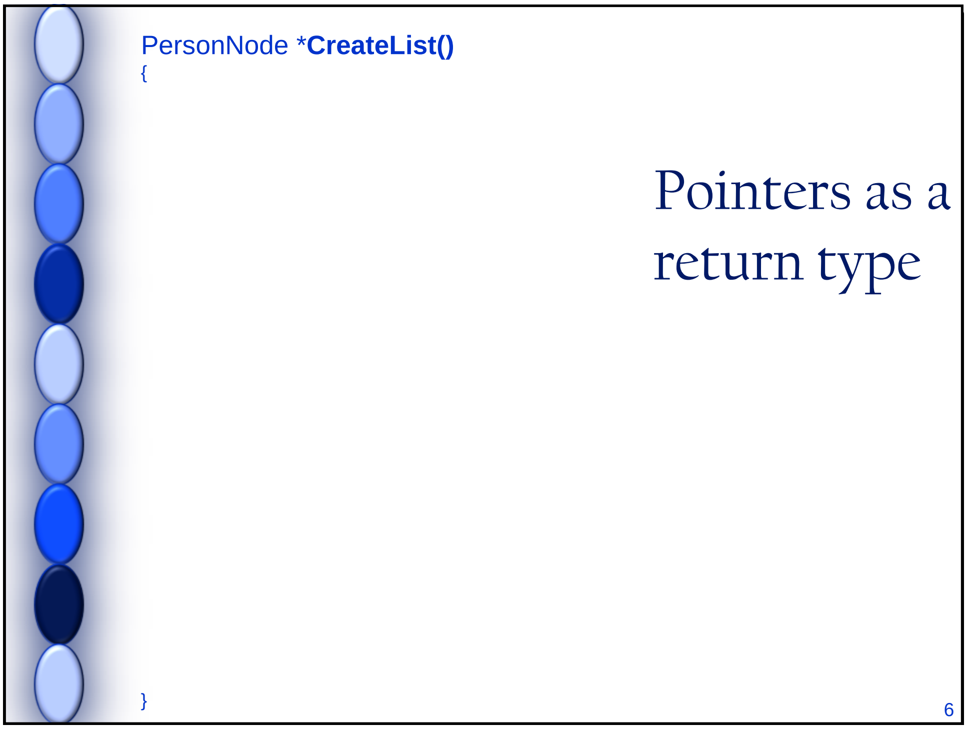
(so it the node that first per is pointing to!)



Pointers as a return type

```
PersonNode *CreateList()
{

  return head;
}
```

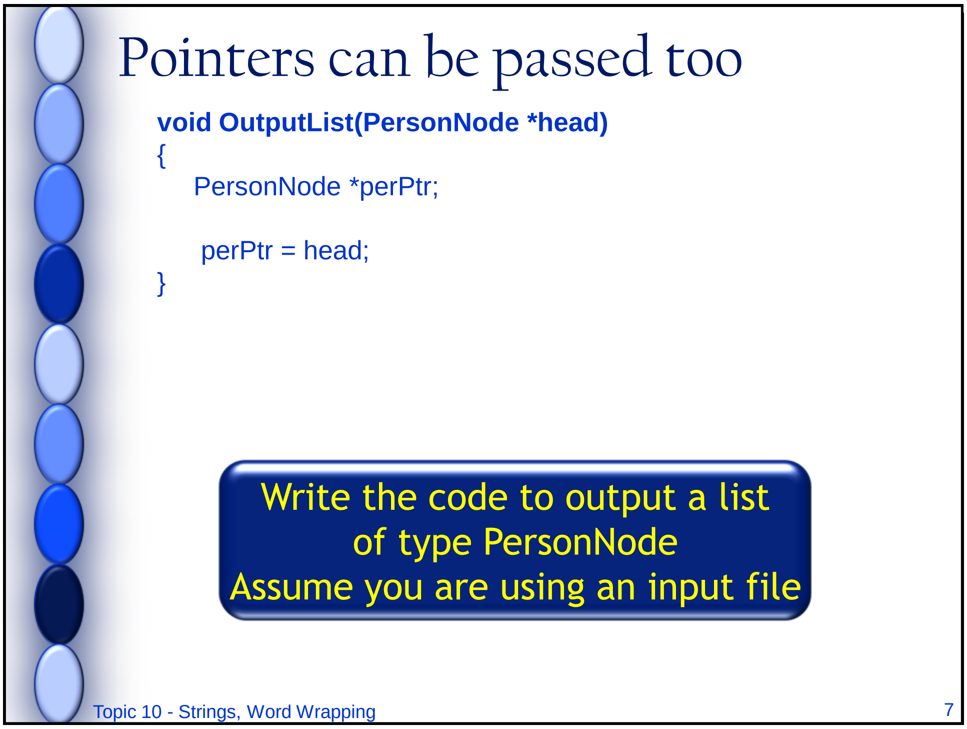
Write the code to create a link-list
of type PersonNode
Assume you are using an input file



```
PersonNode *CreateList()
{
```

Pointers as a return type

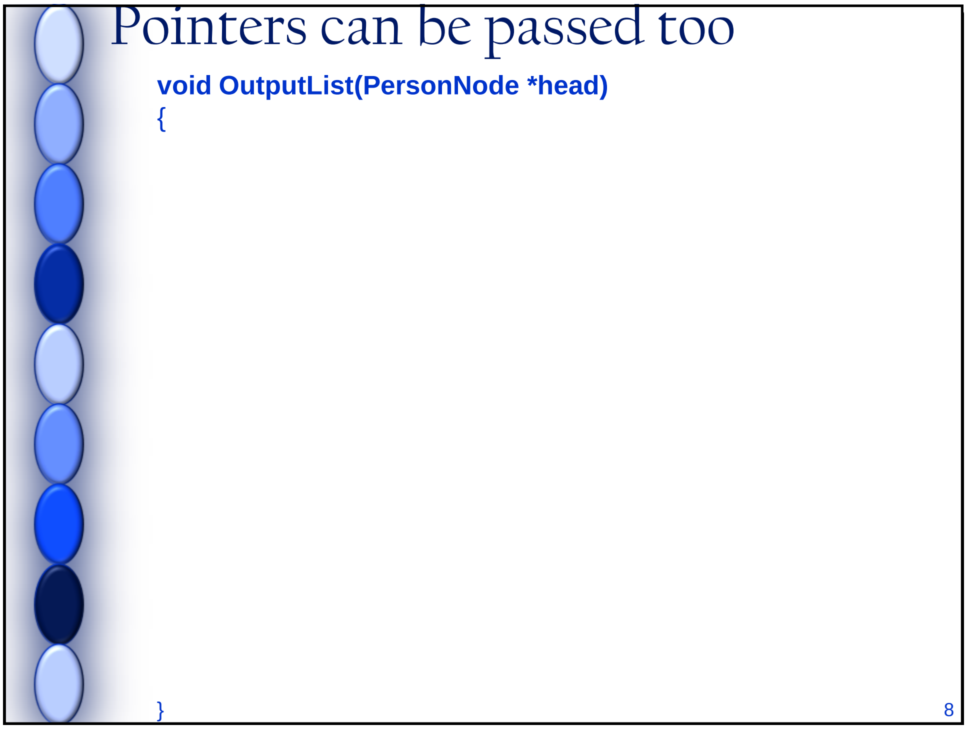
```
}
```



Pointers can be passed too

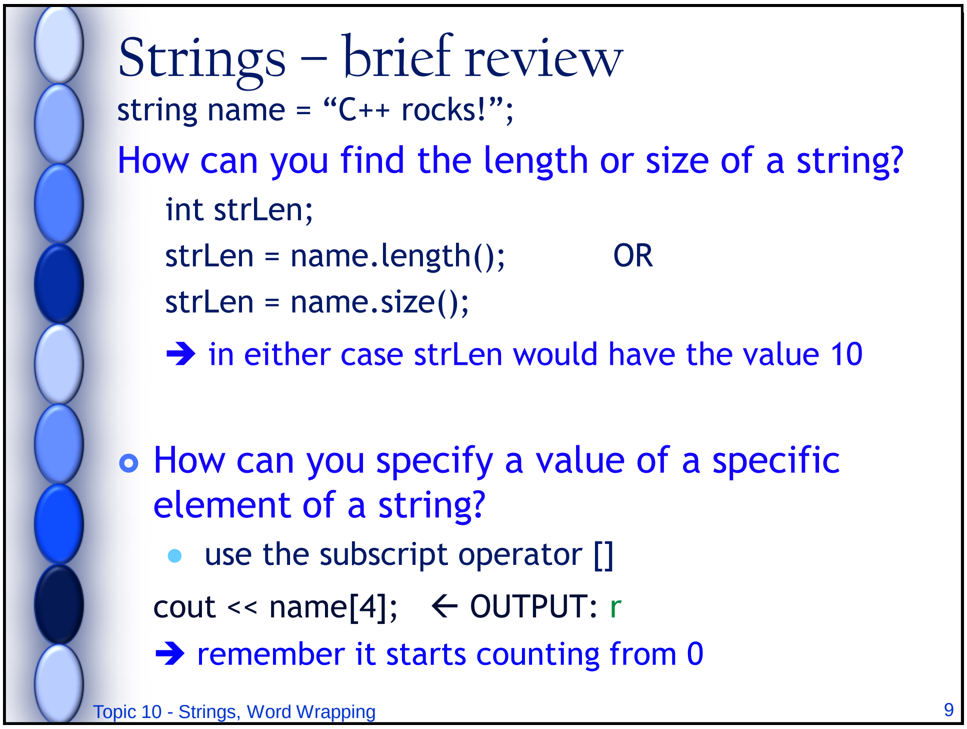
```
void OutputList(PersonNode *head)
{
    PersonNode *perPtr;

     perPtr = head;
}
```

Write the code to output a list
of type PersonNode
Assume you are using an input file



Pointers can be passed too

```
void OutputList(PersonNode *head)
{










}
```



Strings – brief review

string name = “C++ rocks!”;

How can you find the length or size of a string?

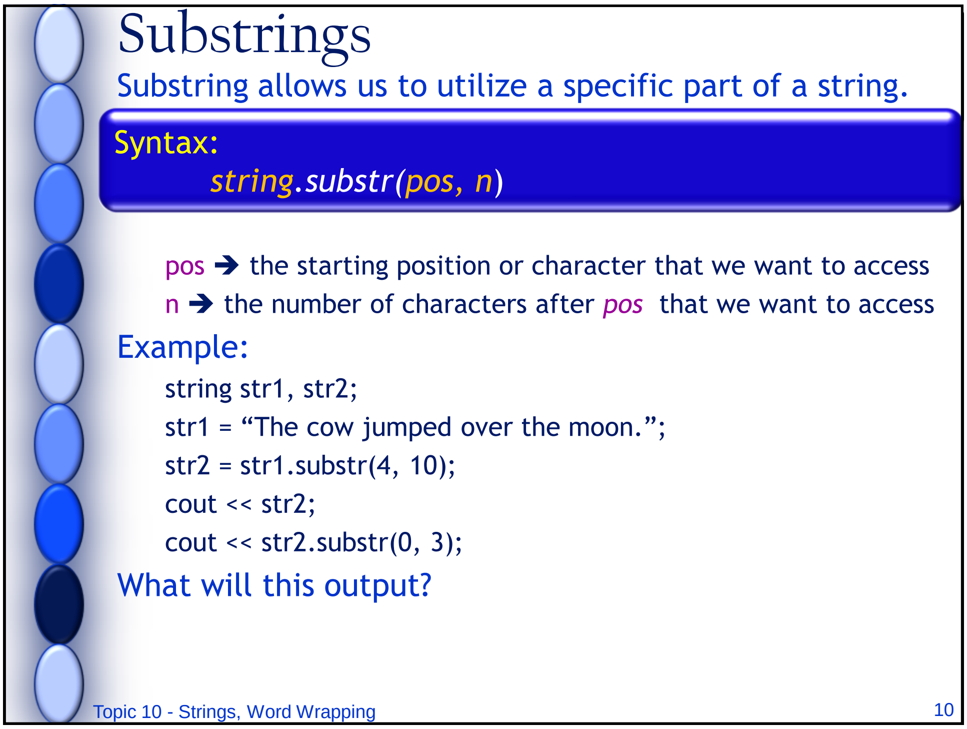
```
int strLen;
strLen = name.length();          OR
strLen = name.size();
```

➔ in either case strLen would have the value 10

- How can you specify a value of a specific element of a string?
  - use the subscript operator []

cout << name[4];   ← OUTPUT: r

➔ remember it starts counting from 0



Substrings

Substring allows us to utilize a specific part of a string.

Syntax:

string.substr(pos, n)

pos ➔ the starting position or character that we want to access

n ➔ the number of characters after *pos* that we want to access

Example:

```
string str1, str2;
str1 = “The cow jumped over the moon.”;
str2 = str1.substr(4, 10);
cout << str2;
cout << str2.substr(0, 3);
```

What will this output?

Parsing a string

- To word wrap:
  - You need to first find the length of the input string
  - You need to have a maximum number of characters you can output per line

  - Loop through each character of the input string
  - Count the number of characters and output a newline when it exceeds the maximum
- How do we avoid starting the new line with a space?
- How do we avoid splicing in the middle of a word?
- How do you know when you hit the end of a word?
  - Add each character to a word → keep track of the word size
  - If the word fits → add it to the line

Word Wrapping

```
RUN a loop from 0 to inputString.length()
    IF inputStr[index] IS NOT EQUAL to ' ' ← CHECK each char
          ADD the char to the word
    ELSE
          IF line.length + word.length > maxLength
              OUTPUT the line
              CLEAR the line
          ADD the word and a space to the line
          CLEAR the word
```

OUTPUT

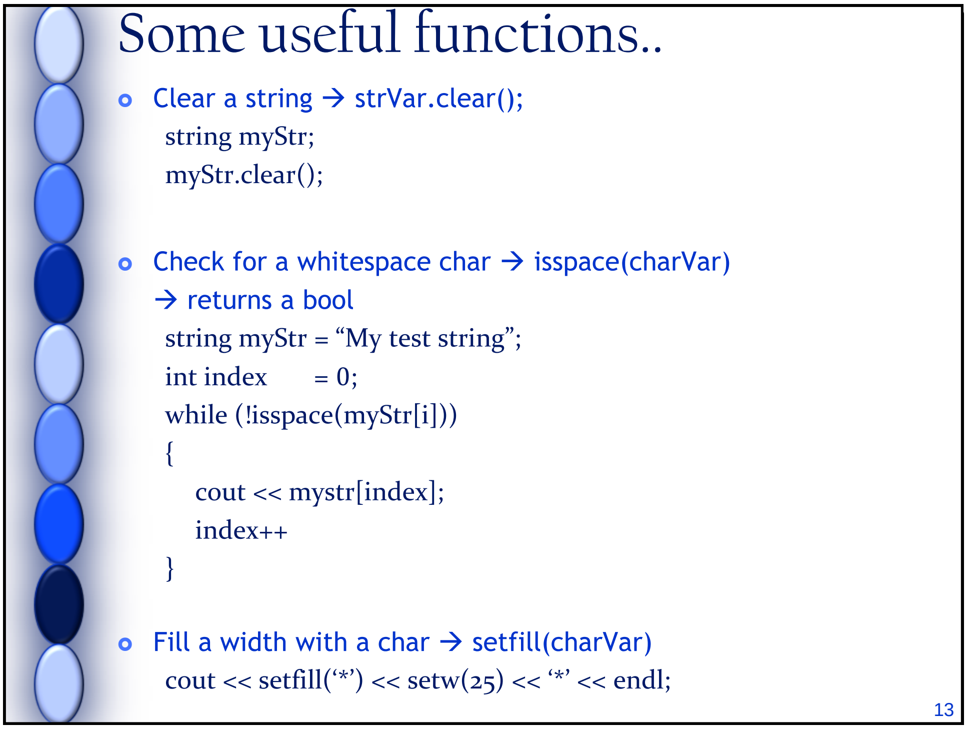
inputStr = "The cow jumped over the moon";

maxLength = 8;

outLineStr

outWordStr

NOTE:
This algorithm doesn't handle the last line/word when necessary
-YOU need to figure out that part of the algorithm



Some useful functions..

- Clear a string → strVar.clear();

```
string myStr;
myStr.clear();
```

- Check for a whitespace char → isspace(charVar)
  → returns a bool

```
string myStr = “My test string”;
int index      = 0;
while (!isspace(myStr[i]))
{
    cout << mystr[index];
    index++
}
```

- Fill a width with a char → setfill(charVar)

```
cout << setfill(‘*’) << setw(25) << ‘*’ << endl;
```