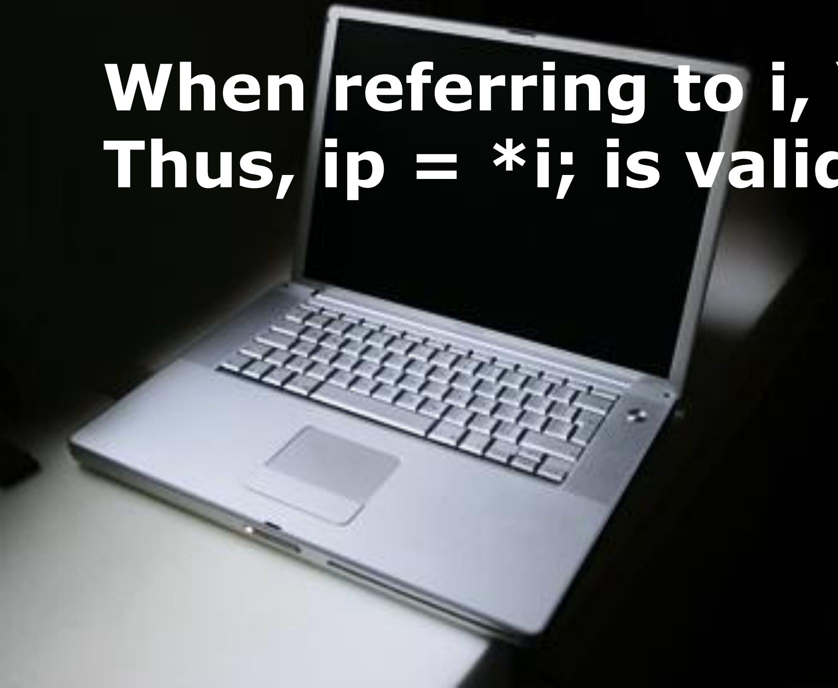
CS211 3-15-12

Pointers:

In delaration, “ * ” meaning the pointer of
 - int *ip; // pointer of integer
 - int *(*i); // pointer of the pointer of integer

They’re different type!
Thus, can’t assign the contents in ‘‘i’’ to ‘‘ip’’.

When referring to i, ‘‘ * ’’ meaning de-reference
Thus, ip = *i; is valid.



More:

int b[][];

# ← some number

b ← referring to 2-D array

b + # ← point to # offset of this 2D array b

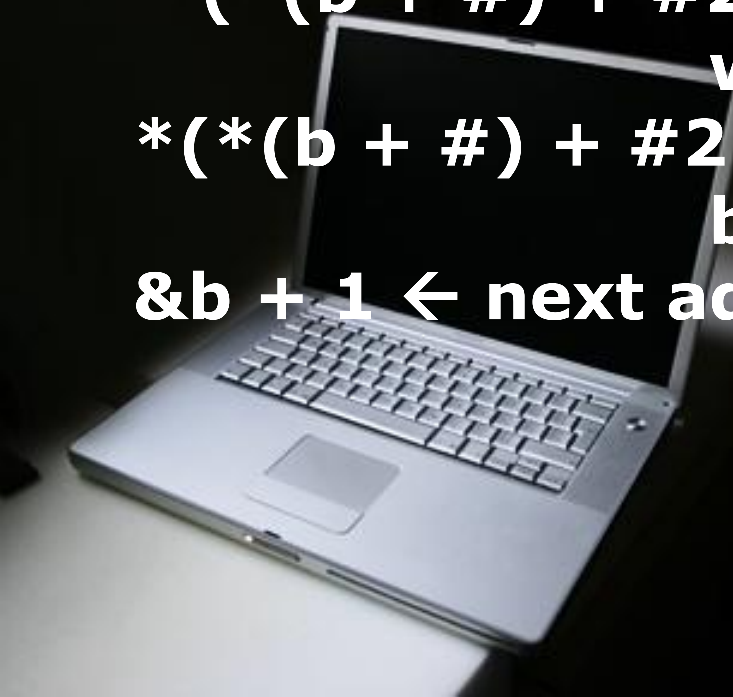
*(b + #) ← the content of # offset of b, which is 1-D array b[#]

*(b + #) + #2 ← point the #2 offset of b[#]

((b + #) + #2) ← content of b[#][#2] which is an integer

((b + #) + #2) + #3 ← #3 add to integer b[#][#2], still an integer

&b + 1 ← next address after 2-D array b



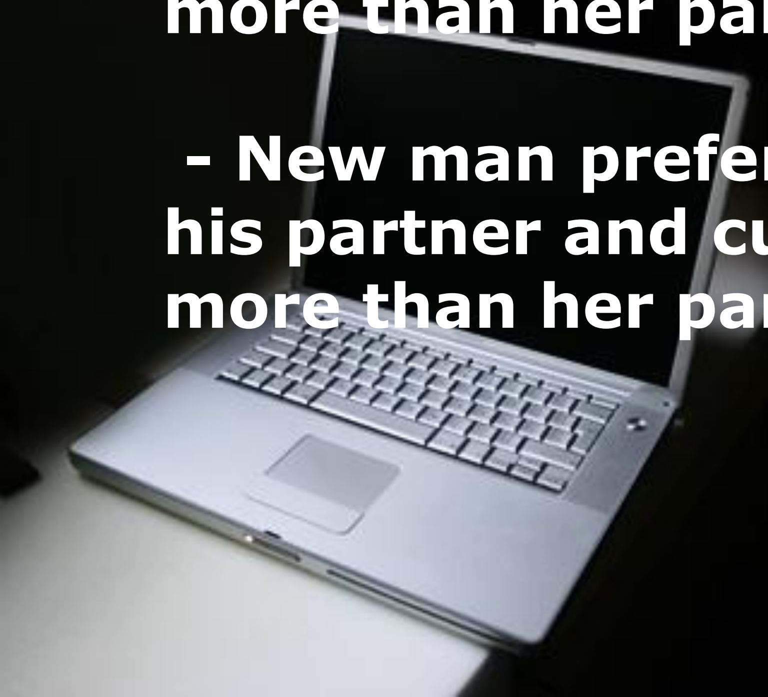
Stable Marriage:

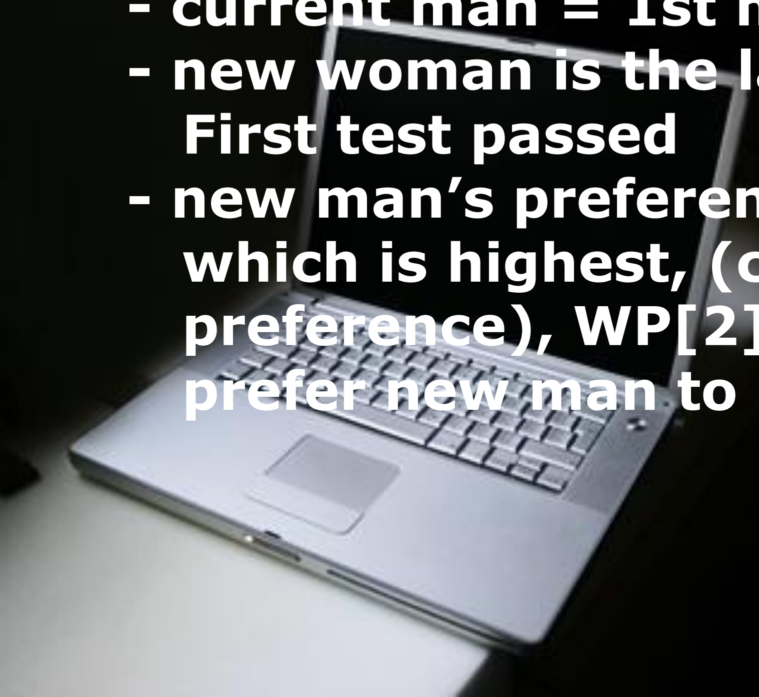
Check against the men and women’s preference ranking tables

- A woman can’t be assigned twice

- Current man prefers new woman more than his partner and this new woman prefer current man more than her partner

- New man prefer the current woman more than his partner and current woman prefer new man more than her partner





Example:

Match[3] = {2, 0, 1}
Is ‘‘ 2 0 1 ’’ a stable match to following preference tables?
MP[3][3] = {{0, 2, 1}, {0, 2, 1}, {1, 2, 0}}
WP[3][3] = {{2, 1, 0}, {0, 1, 2}, {2, 0, 1}}
No!
Check the test!

- **Every woman assigned once ← pass**
- **last pair ← new man = 3rd man, new woman = 2nd woman**
  Look thru all pairs.
  - **current man = 1st man, current woman = 3rd woman**
  - **new woman is the last preference on current man’s list**
    First test passed
  - **new man’s preference to current woman is MP[2][2], which is highest, (check whether is mutually higher preference), WP[2][2] > WP[2][0] ← current woman prefer new man to her partner as well.**

Not Stable!

ok() function:

```
bool ok(int match[], int col) {
  int current_man, current_woman, new_man, new_woman;
  new_man = col;
  new_woman = match[col];

  //check whether new woman has previously assigned
  for (int i = 0; i < col; i++)
     if (match[i] = new_woman) return false;

  //check for all current men and current women
  //whether new man and new women introduce an instability
  for (current_man = 0; current_man < col; current_man++){
     current_woman = match[current_man];
     if (____________&&_______________) return false;
     if (____________&&_______________) return false;
  }
  return true;
}
```