POINTERS AND DYNAMIC MEMORY ALLOCATION (REVIEW)

Problem Solving with Computers-II

https://ucsb-cs24-sp17.github.io/

Read the syllabus. Know what's required. Know how to get help.

CLICKERS OUT – FREQUENCY AB

Announcements

- Midterm on Wed 04/26
- Study session today (04/23) from 7pm to 9pm in HFH 1132

Pointers

- Pointer: A variable that contains the address of another variable
- Declaration: *type* * pointer_name;

```
int *p;
```

How do we initialize a pointer?

How to make a pointer point to something

```
int *p;
int y;
```

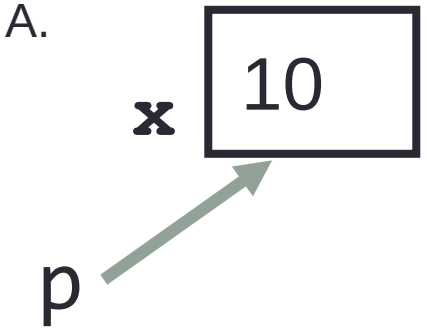
To access the location of a variable, use the address operator ‘&’

Tracing code involving pointers

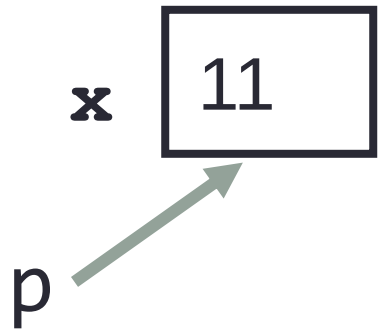
```
int *p, x=10;
p = &x;
*p = *p + 1;
```

Q: Which of the following pointer diagrams best represents the outcome of the above code?

A.



B.

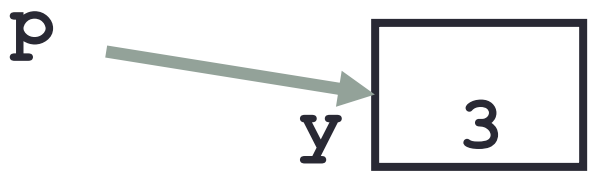


C. Neither, the code is incorrect

Dynamic memory: Make p point to an int on the heap

```
int *p;
int y;
p = &y;
```

Two ways of changing the value of a variable



Change the value of y directly:

Change the value of y indirectly (via pointer p):

Pointer examples: Trace the code

```
int x=10, y=20;

int *p1 = &x, *p2 =&y;

p2 = p1;

int **p3;

p3 = &p2;
```

Pointer assignment

```
int *p1, *p2, x;
p1 = &x;
p2 = p1;
```

Q: Which of the following pointer diagrams best represents the outcome of the above code?

A.

B.

C. Neither, the code is incorrect

Mechanics of function calls on the run-time stack

```
double getAverage(int * sc, int len){
  double sum=0;
  for (int i=0; i<len; i++){
        sum+=sc[i];
  }
  return (sum/len);
}

int main(){
  int scores[5]={65, 85, 97, 75, 95};
  int len = 5
  double avg_score;
  avg_score = getAverage(scores,len);
  cout<< avg_score;
}
```

Dynamic memory allocation

- To allocate memory on the heap use the ‘new’ operator
- To free the memory use delete

```
int *p= new int;
delete p;
```

Dangling pointers and memory leaks

- Dangling pointer: Pointer points to a memory location that no longer exists
- Memory leaks (tardy free) Memory in heap that can no longer be accessed

Q: Which of the following functions results in a dangling pointer?

```
int * f1(int num){
    int *mem1 =new int[num];
    return(mem1);
}
```

```
int * f2(int num){
     int mem2[num];
     return(mem2);
}
```

A. f1

B. f2

C. Both

Rewrite the code using dynamic arrays

```
double getAverage(int * sc, int len){
  double sum=0;
  for (int i=0; i<len; i++){
        sum+=sc[i];
  }
  return (sum/len);
}

int main(){
  int scores[5]={65, 85, 97, 75, 95};
  int len = 5
  double avg_score;
  avg_score = getAverage(scores,len);
  cout<< avg_score;
}
```

Write the declaration of the allocate space function

```
int main(){
  int * scores, size_t n;
  allocate_space(scores, n)
  // scores should point to a dynamic array of size n, where n is input by the user

}
```

DEMO

- Dynademo.cxx (Program to demo dynamic arrays)
- How to use valgrind to detect memory leaks
- Debugging segfaults with gdb and valgrind

Next time

- Chapter 4 (contd): Bag class with dynamic arrays, intro to linked-lists