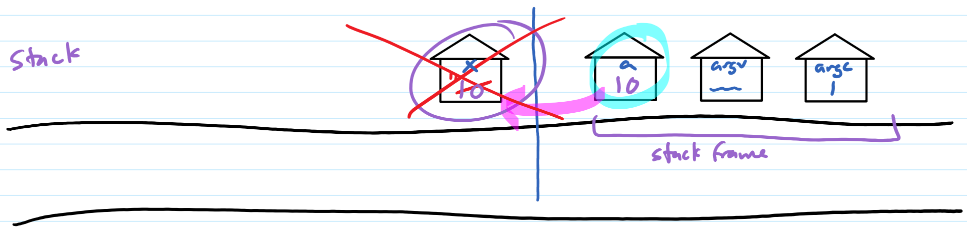
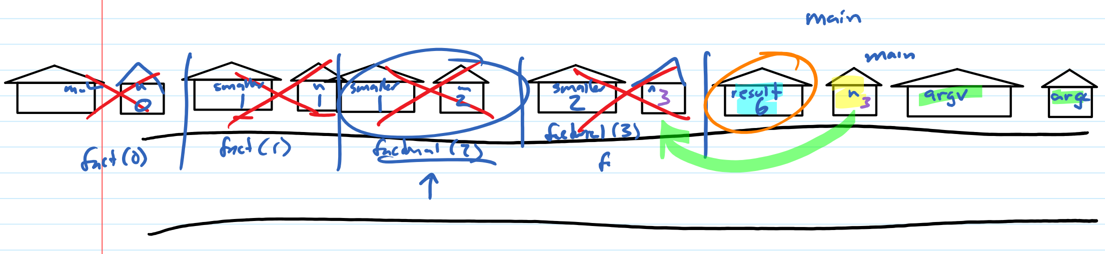


```
int main(int argc, char *argv[]) {
  int a = 10;
  increment(a);
  printf("a=%d\n", a);
}
```

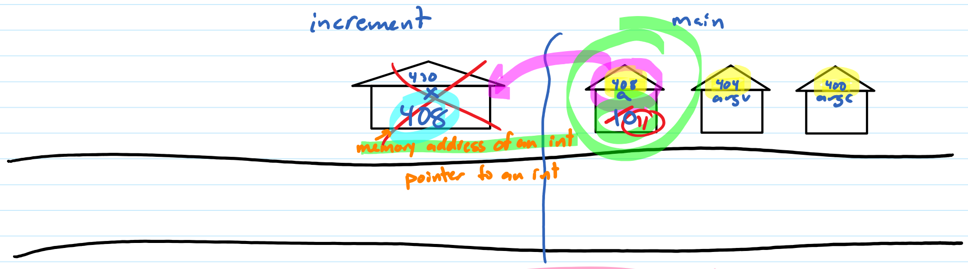
```
int increment(int x) {
  x = x + 1;
}
```



```
int main(int argc, char *argv[]) {
  int n;
  scanf("%d", &n);
  long result = factorial(n);
  printf("%d! = %ld\n", n, result);
}
```

6

```
long factorial(int n) {
  if (n == 0)
      return 1;
  else {
      long smaller = factorial(n - 1);
      return n * smaller;
  }
}
```



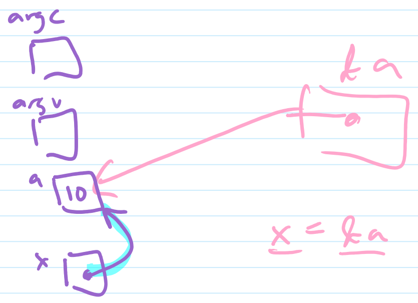
```
int main(int argc, char *argv[]) {
  int a = 10;
  increment(&a);
  printf("a=%d\n", a);
}
```

11

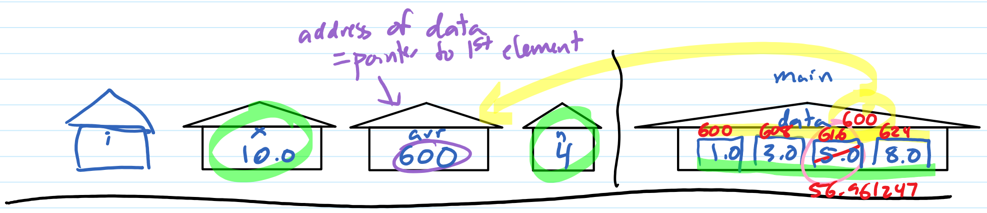
```
void increment(int *x) {
  *x = *x + 1;
}
```

dereference

x = x+1



address of data
= pointer to 1st element



```
int main(int argc, …)                         void add_all(int n, double *arr, double x) {
{                                               for (int i = 0; i < n; i++) {
  double data[] = {1.0, 3.0, 5.0, 8.0};             arr[i] += x;
  // …                                            }
  add_all(4, data, 10.0);                     }
  data[2] = 56.961247;
  //…
}
```

$*(data + 2)$ C

$*(data + 2 * \underbrace{sizeof}_{\text{\# of bytes for type}}(double))$ human

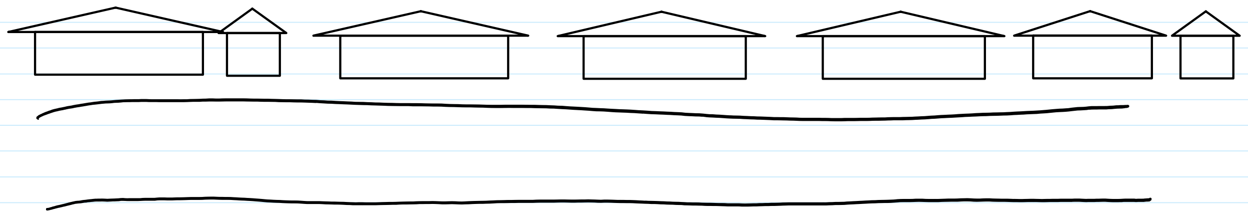
$600 + 2 * 8$

$= 616$

$*(arr + i)$ C

$*(arr + i * (sizeof(double)))$ human

```
typedef struct _rectangle
{
  int left;
  int top;
  int width;
  int height;
} rectangle;

rectangle enlarged(rectangle r, double scale)
{
  rectangle result = {r.left, r.top, r.width * scale, r.height * scale};
  return result;
}

int main(int argc, char *argv[])
{
  rectangle home = {20, 40, 15, 25};
  rectangle bigger = enlarged(home, 2.0);

  printf("%d %d %d %d\n", home.left, home.top, home.width, home.height);
  printf("%d %d %d %d\n", bigger.left, bigger.top, bigger.width, bigger.height);

}
```



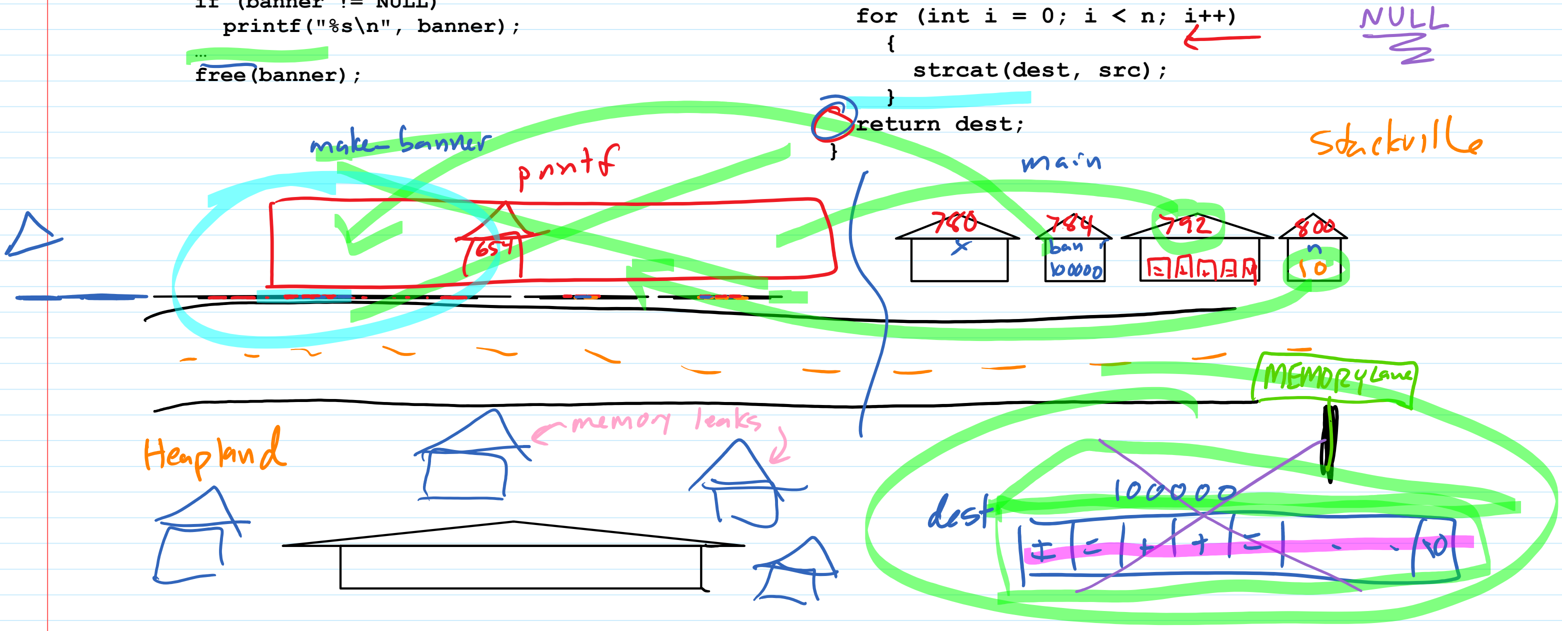
#include <stdlib.h>   malloc
                        free

```
int main() {
  // …

  int n = 10;
```

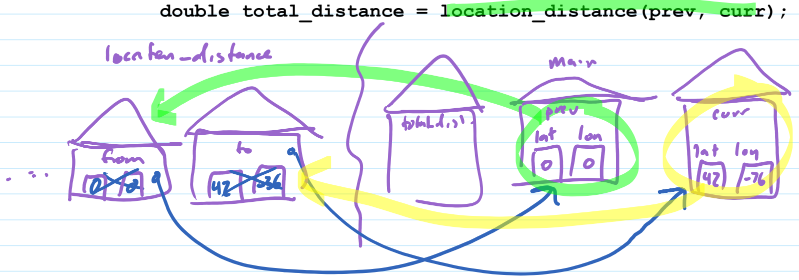
```
char *make_banner(char *src, int n)
{
  int len = strlen(src);
  char *dest = malloc((len * n + 1) * sizeof(char));
```

```
int main() {
  // …

  int n = 10;
  char piece[] = "=++=";
  char *banner = make_banner(piece, 2);
  if (banner != NULL)
    printf("%s\n", banner);
  …
  free(banner);
```

```
char *make_banner(char *src, int n)
{
  int len = strlen(src);
  char *dest = malloc((len * n + 1) * sizeof(char));
if (dest == NULL) return NULL;
  strcpy(dest, "");
  for (int i = 0; i < n; i++)
    {
      strcat(dest, src);
    }
  return dest;
}
```



```
location prev;
  location curr;

  double total_distance = location_distance(prev, curr);
```



location_distance
from
to
42
-86
total_dist
main
prev
lat
lon
0
0
curr
lat
lon
42
-76