```
int p =2;
int main()
{
   int num;
}
```

early binding

memory reserved at compiletime.

Array can be implemented

① Statically — can't be Shrink | grown at runtime.

② Dynamically — can be Shrinked | grown at runtime

controlled

programmer

allocate

free deallocate

heap — request

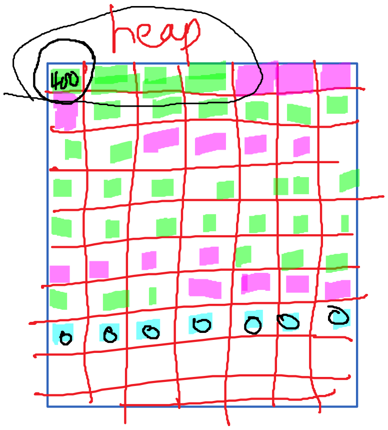
malloc | calloc

fail

NULL

success.

address of a byte from heap



size_t — unsignedint.

1st call → malloc(4)

~~void*~~ malloc (sizeof(int))

2nd call.
malloc(8).

400

typecast.

realloc

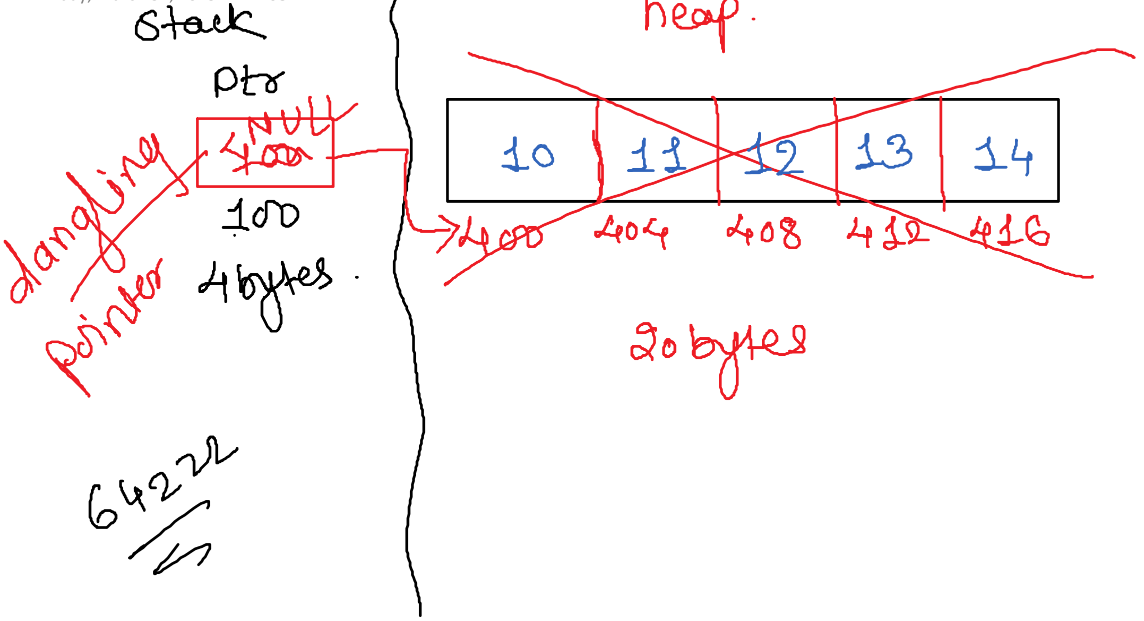
calloc(5, sizeof(int)) /

$\frac{5*4}{20}$.

requested memory use by default assigned with 0

default value garbage.

malloc ($\underbrace{\text{sizeof(int)} * 5}_{20}$) ;

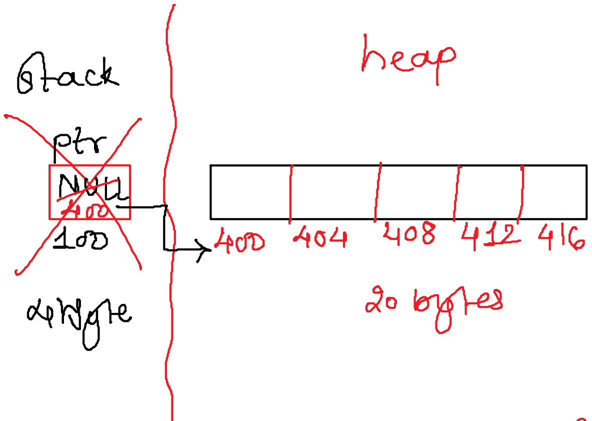


free(ptr);

free(400)

Friday, March 31, 2023 4:28 PM

```
void test()
{
    int *ptr = NULL;
    ptr = (int *)malloc(sizeof(int)*5);
}
int main()
{
    test();
}
```



on exit of function memory given for ptr (4 bytes) will be altomatically released and bcoz of that there is no pointer to access dynamic meromey

hence we can not use dynamic memory nor even release it. this situation is called memory leakage