Static v/s Dynamic
Memory Allocation

Party
↳ chairs:
1) Arrange → how many guest → Static Alloca
2) No idea → situation → Dynamic Allocation.

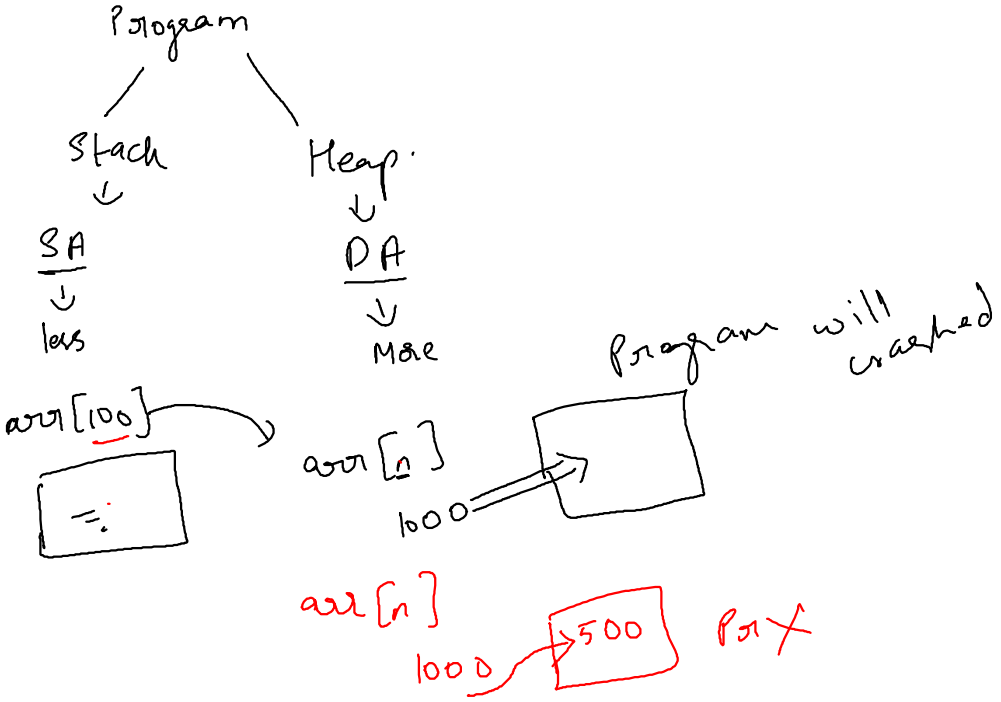
```
int n;
cin>> n;
int arr[n];
```

10
50
1000

Runtime

compile

int arr[1000]; ✓



↳ Stack
↳ Memory hai that will be fixed at compile time

5
10
100

int arr[5];

Heap

→ variable size

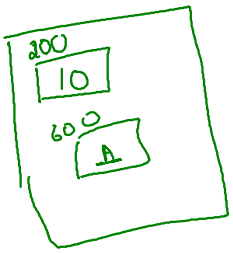
int arr[50]; → stack.

x.

Heap.

new

new int(10);



heap

pointer → address

int x = 10

int * ptr = & x;

```
int *ptr = new int;
```

cout << *ptr ; 10 //.

int *ptr = new char;

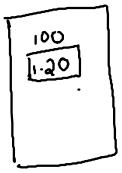
BTS

```
int *ptr = new int(20);
```

ptr => 100

*ptr => 20.

Stack



Heap

int => 4 bytes
ptr = 8 bytes } 12 bytes

*ptr → address

int *ptr
char *ptr } 8 bytes

32 → 4

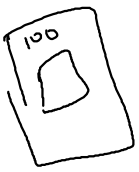
64 → 8

char *ptr = new char

↓ ↓

8 bytes 1 bytes

9 bytes.



Heap

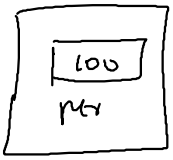
DMA → Heap.

```
int n;
cin >> n;
int *ptr = new int[n];
```

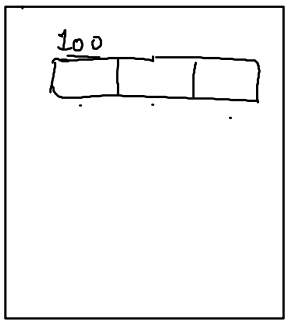
→ Runtime

-> Runtime
-> Badaa
=> code.

int *ptr = new int[3];



Stack



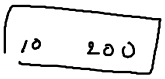
Heap

8 bytes + 12 bytes.
= 20 bytes.

int arr[1000000] // May crash chota

new int[100000] // large -> heap

Example

n => 

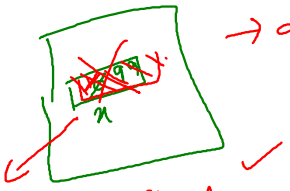
+ -> o/p //.

Major Difference S & H.

Stack

```
for(int i=0; i<100; i++) {
    int x = i;
}
```

100



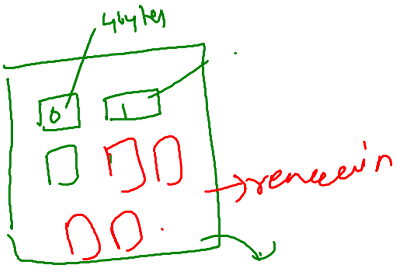
-> automatically

4 bytes. stack.

Heap.

```
for(int i=0, i<100; i++){
    int *x = new int(i);
}
```

8 bytes.



4 x 100 = 400 bytes.

delete x

Chandi -> memory -> program.

int *arr = new int[50];

delete [] arr;

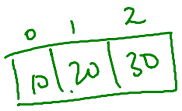
SMA -> Stack -> Compile -> Fixed -> Fast -> Auto matic

DMA -> Heap -> Runtime -> Flexible -> Slow -> Manual delete.

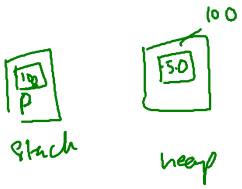
```
int arr[5];
```

```
int *arr = new int[n];
```

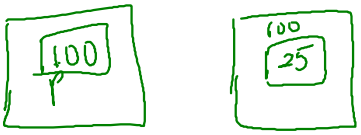
Q1: What is the output?
```
int arr[3] = {10, 20, 30};
cout << arr[1];
```
A) 10
B) 20
C) 30
D) Garbage



Q2: What is the output?
```
int *p = new int(50);
cout << *p;
```
A) 0
B) 50
C) Address
D) Error

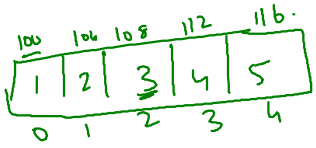


Q3: What is the output?
```
int *p = new int;
*p = 25;
cout << *p;
```
A) 0
B) 25
C) Garbage
D) Error

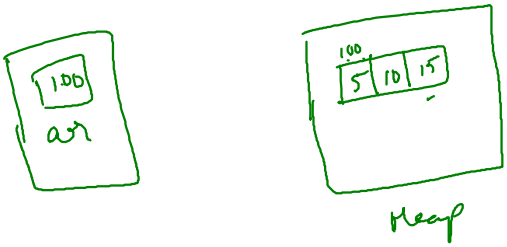


Q4: What is the output?
```
int arr[5] = {1, 2, 3, 4, 5};
int *p = arr;
p += 2;
cout << *p;
```
A) 2
B) 3
C) 4
D) Garbage

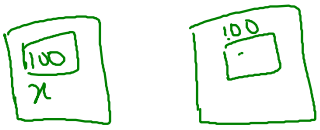
p = p + 2



Q5: What is the output?
```
int *arr = new int[3];
arr[0] = 5;
arr[1] = 10;
arr[2] = 15;
cout << arr[2];
```
A) 5
B) 10
C) 15
D) Garbage



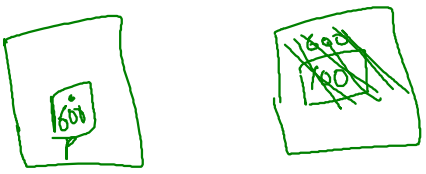
Q6: What is the output?
```
int *x = new int;
cout << *x;
```
A) 0
B) Garbage
C) Error
D) Address



Q7: What is the output?

```
int *p = new int(100);
delete p;
cout << *p;
```

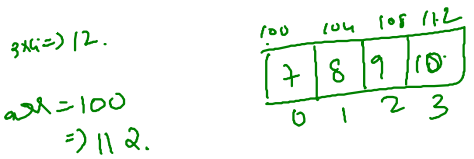
A) 100
B) 0
C) Undefined / Garbage
D) Error



Q8: What is the output?

```
int arr[4] = {7, 8, 9, 10};
cout << *(arr + 3);
```

A) 7
B) 8
C) 9
D) 10



Q9: What is the output?

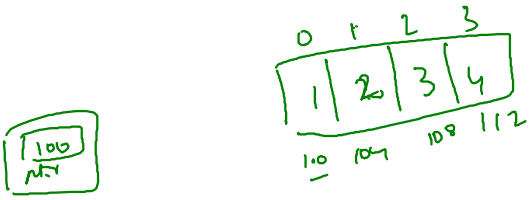
```
int *p = new int(99);
int *q = p;
*q = 11;
cout << *p;
```

A) 11
B) 99
C) 0
D) Error

Q10: What is the output?

```
int arr[] = {1, 2, 3, 4};
int *ptr = arr;
cout << ptr[1];
```

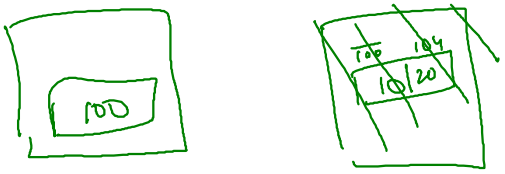
A) 1
B) 2
C) 3
D) Garbage



Q11: What is the output?

```
int *arr = new int[2]{10, 20};
delete[] arr;
cout << arr[1];
```

A) 10
B) 20
C) Garbage / Undefined
D) Error



Q12: What is the output?

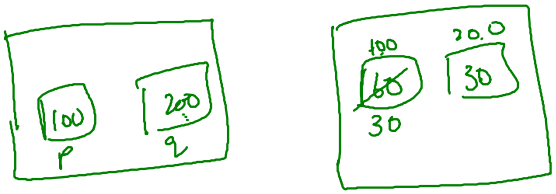
```
int *p = new int(70);
cout << &p;
```

A) 70
B) Garbage
C) Address of pointer variable
D) Address of 70

Q13: What is the output?

```
int *p = new int(60);
int *q = new int(30);
*p = *q;
cout << *p;
```

A) 60
B) 30
C) 0
D) Error



Q14: What is the output?

```
int *arr = new int[4];
arr[0] = 3;
arr[1] = 6;
arr[2] = 9;
arr[3] = 12;
cout << *(arr + 2);
```

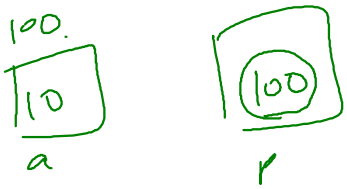
A) 6
B) 9
C) 12
D) Garbage

100+8

Q15: What is the output?

```
int a = 10;
int *p = &a;
cout << sizeof(p);
```

A) 2
B) 4
C) 8 (on 64-bit)
D) 10



Q16: What is the output?

```
int *arr = new int[5];
for(int i = 0; i < 5; i++) {
    arr[i] = i * 2;
}
cout << arr[4];
```

A) 4
B) 8
C) 10
D) 2

H/w.

Q17: What is the output?

```
int arr[] = {100, 200, 300};
cout << *(arr + 1);
```

A) 100
B) 200
C) 300
D) Error

H/w

Q18: What is the output?

```
int *p = new int(5);
cout << p;
```

A) 5
B) Garbage
C) Address
D) Error

H/w

Q19: What is the output?

```
int a = 5;
int *p = &a;
delete p;
cout << a;
```

A) 5
B) Garbage
C) Error
D) Crash

Q20: What is the output?

```
int *arr = new int[3]{1, 2, 3};
cout << *(arr + 2);
```

A) 1
B) 2
C) 3
D) Garbage