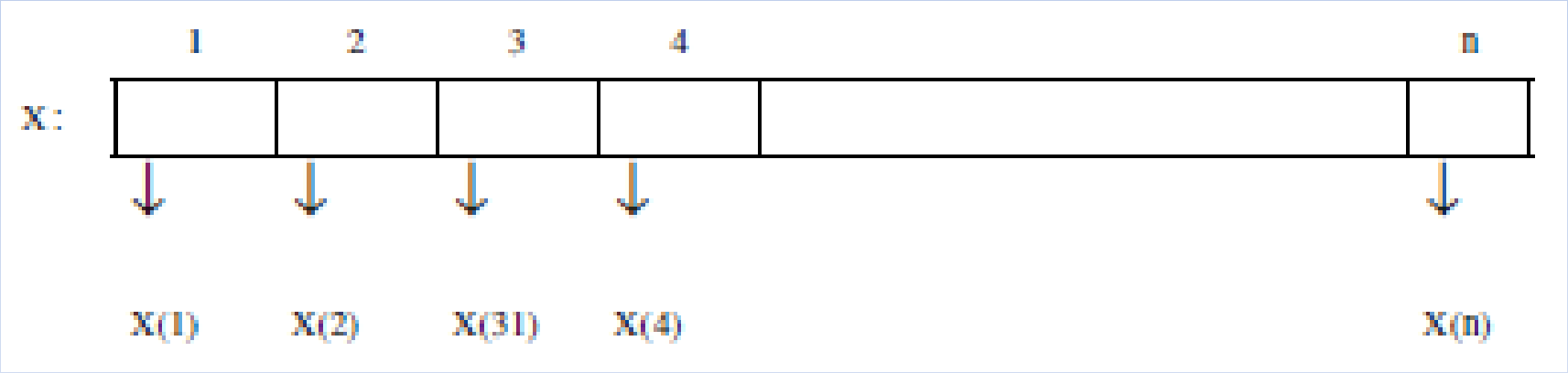
Array, Structure, and Pointer

Farah Zakiyah Rahmanti, M.T

Static Array

- Array is used to handle a lot of data with same type.



Fill Array

```
int x[10];
scanf("%d", temp);

for (i=0; i<temp; i++)
  scanf("%d", x[i]);
```

Fill Array – Avoid Same Data

```
scanf(“%d”, &n);
k = 1;

do
   scanf("%d", &b);
   isAvailable=0;

   for(i=1; i<k-1; i++)
        if(b == x[i])
               isAvailable = 1;
        if(! isAvailable) {
               x[k] = b;
               k = k + 1;
        }
        else
               printf("data is already exist\n");
while (k > n)
```

Looking For The Largest Data (max)

```
max = x[1];


for(i=2; i<n; i++)
  if (x[i] > max)
     max = x[i];
```

Looking For The Smallest Data (min)

```
min = x[1];


for(i=2; i<n; i++)
  if (x[i] < min)
     min = x[i];
```

Looking For The Largest (max) and The Smallest Data (min)

```
max = x[1];
min = x[1];

for(i=2; i<n; i++)
  if (x[i] > max)
     max = x[i];
  else if(x[i] < min)
     min = x[i];
```

Static & Dynamic Variable

- Static Variable :
- A variable position in permanent memory during running the program, and **can not be altered** thus that the size of the required memory by the program is **static**.

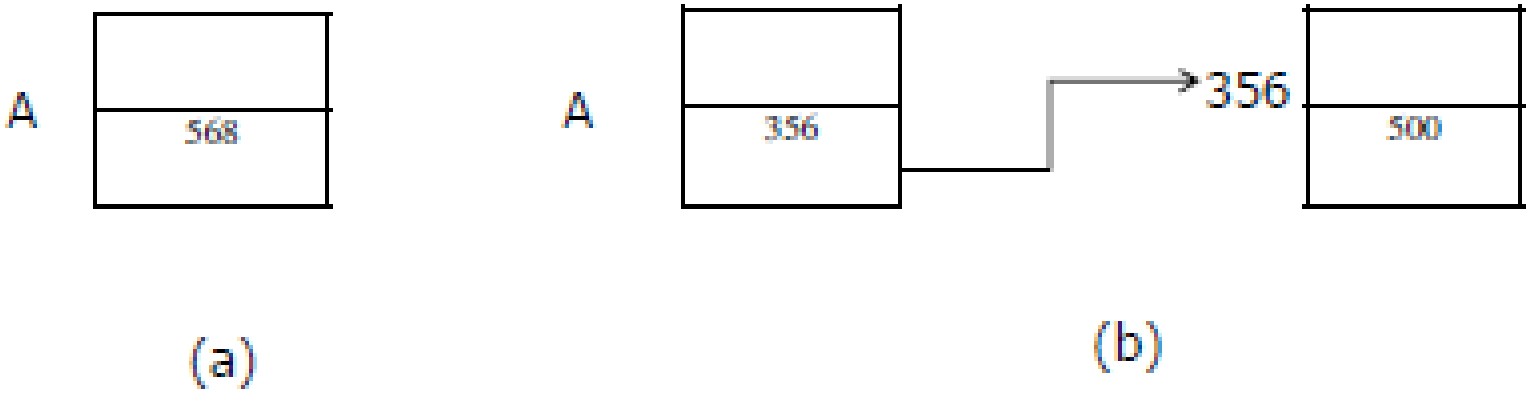
- Dynamic Variable :
- A variable is allocated when it's required, and **can be deleted** when the program is running, thus the size of the required memory by the program is **dynamic**.

Pointer

- Pointer is dynamic data structure , declared variable refers to the location of a specific memory address in RAM.

- Pointer variable is not a value but contains a specific memory address.

Pointer Illustration

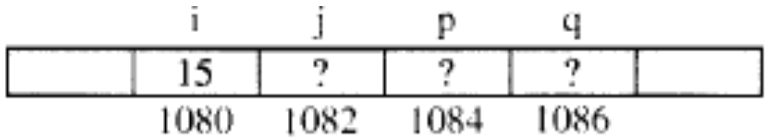


(a) Variable A is static variable, contain static value

(b) Variable A is dynamic variable, contain a specific address (356), which designated address contain a value (500). This designated value is called node.

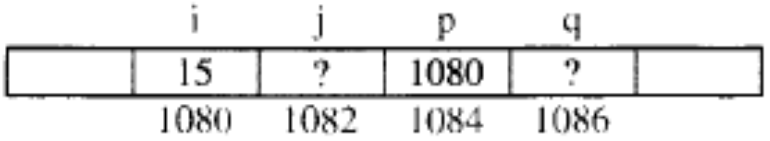
Pointer - Example

```
int i = 15, j,
        *p, *q;
```

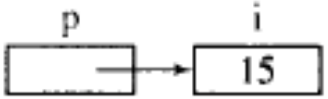


(a)

```
p = &i;
```

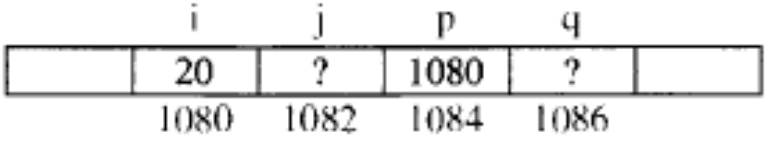


(b)

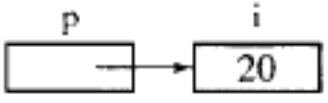


(c)

```
*p = 20;
```

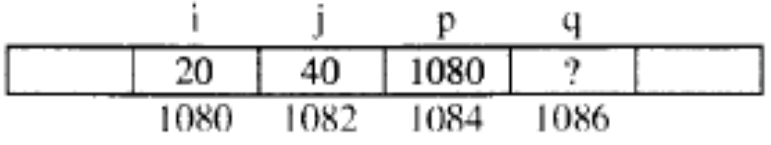


(d)



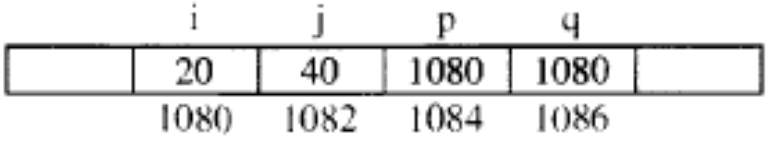
(e)

```
j = 2 * *p;
```

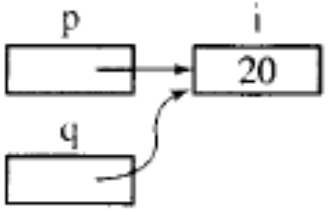


(f)

```
q = &i;
```



(g)



(h)

Structure

- Structure is used when it is needed to process the data with a variety data types, but we still want it to reference the data as a single entity.

Static Structure

```
struct student {
  char name[30];
  float marks;
} student1,student2;
int main() {
  struct student student3;
  char s1[30];float f;
  scanf ("%s", name);
  scanf (" %f", & f);
  student1.name = s1;
  student2.marks = f;
  printf ("Nama : %s \n", student1.name);
  printf (" Marks are %f \n", student2.marks);
  return 0;
}
```

Dynamic Structure

```
typedef struct node *stack;
struct node {
  int data;
  stack next;
};
stack top;
```

Dynamic Structure Allocation

```
typedef struct telm *addr;
struct telm {
        int data;
        addr next;
}elm;
typedef addr stack;
```

```
Allocation addr t :
t = (addr)malloc(sizeof(elm));

Deallocation addr t :
free(t);
```

Stack Implementation

```
typedef struct node *stack;
struct node {
        int data;
        stack next;
};
stack top,bottom;
Int A,B,C;
```

```
top = (struct node*)malloc(sizeof(struct node));
bottom = (struct node*)malloc(sizeof(struct node));

free(top);
free(bottom);
```

Illustration

Linked list