Multiple Strings and Sorting with an Array of Pointers

Supplementary Slides to Lecture 7

Yakup Genc

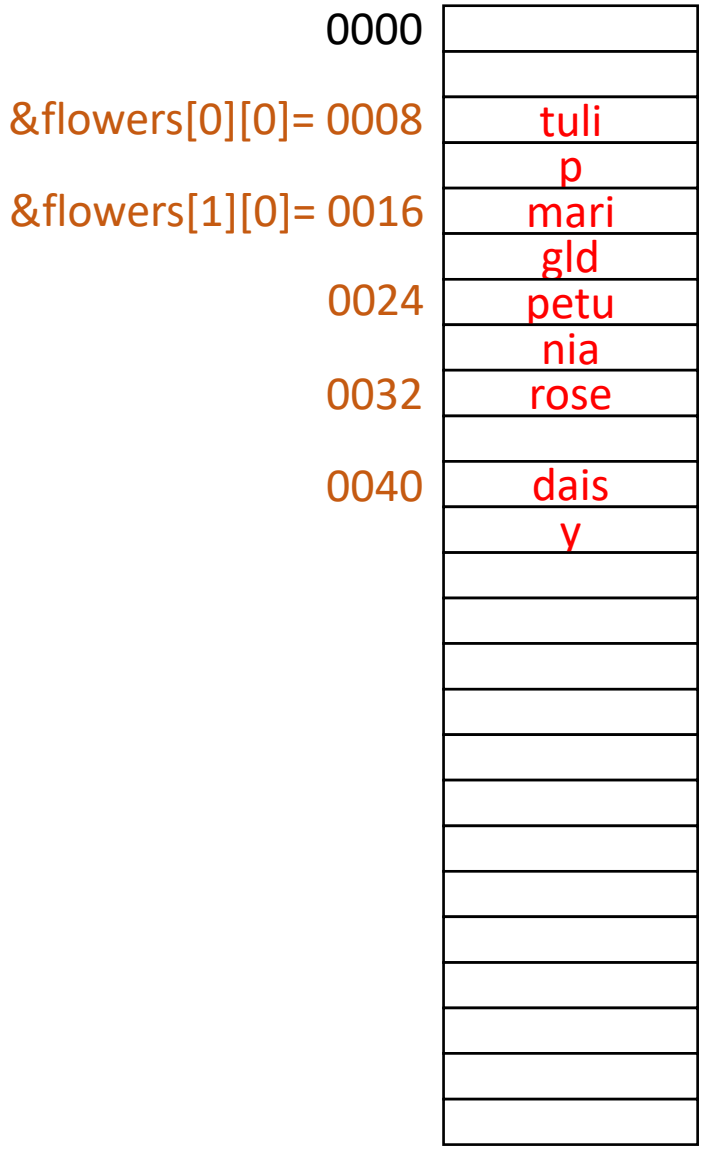
```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

```
Before sorting:
   FFFFCBA0: tulip
   FFFFCBAA: marigld
   FFFFCBB4: petunia
   FFFFCBBE: rose
   FFFFCBC8: daisy
 After sorting:
   FFFFCBC8: daisy
   FFFFCBAA: marigld
   FFFFCBB4: petunia
   FFFFCBBE: rose
   FFFFCBA0: tulip
```

```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Initialization of the string array…

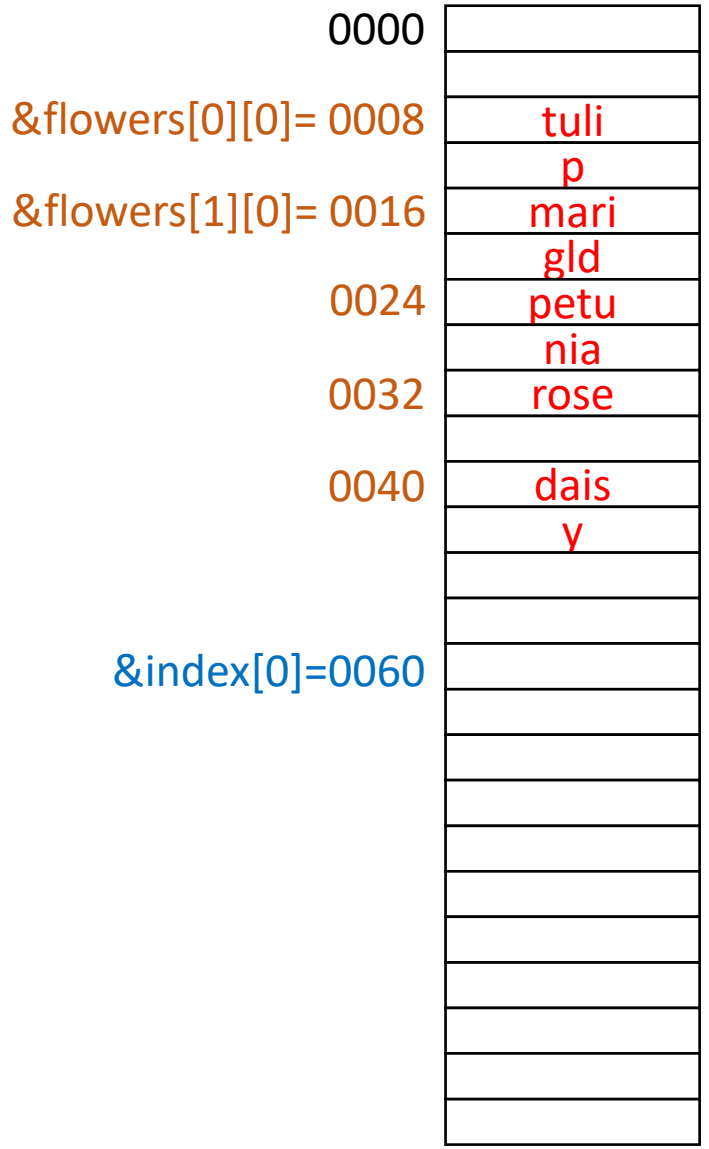
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "mari", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Declaration of the address array

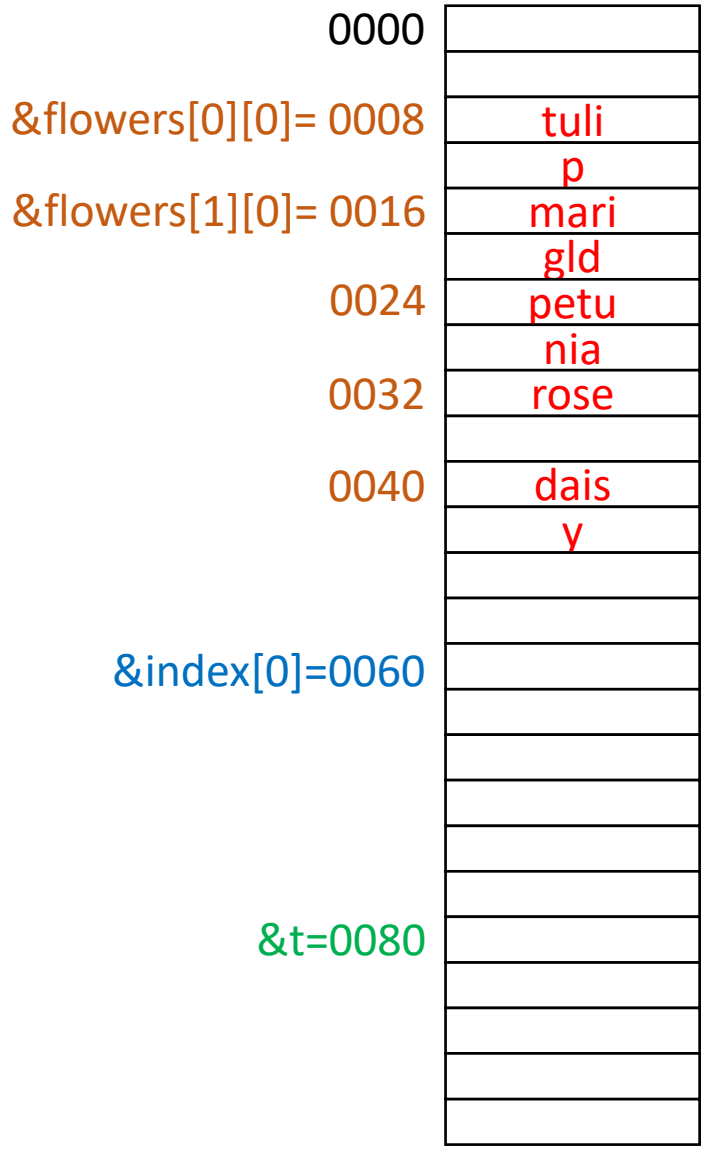
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tuli", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Declaration of temporary pointer for swap…

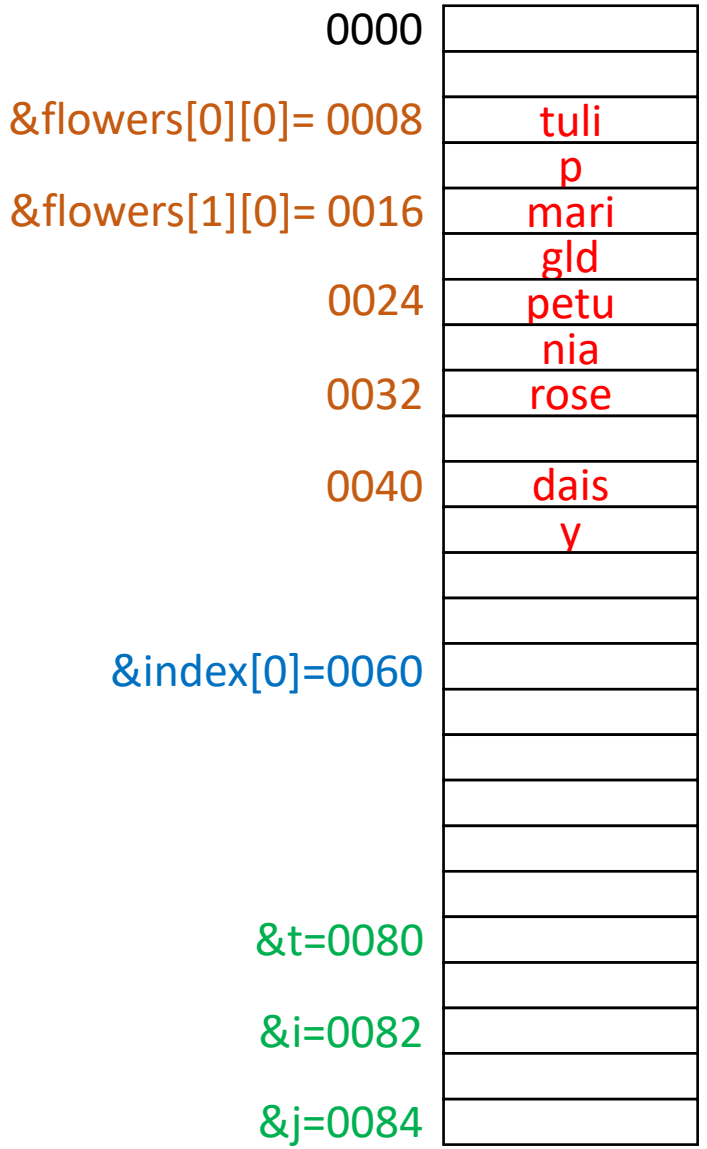
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigold", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Declaration of counter variables

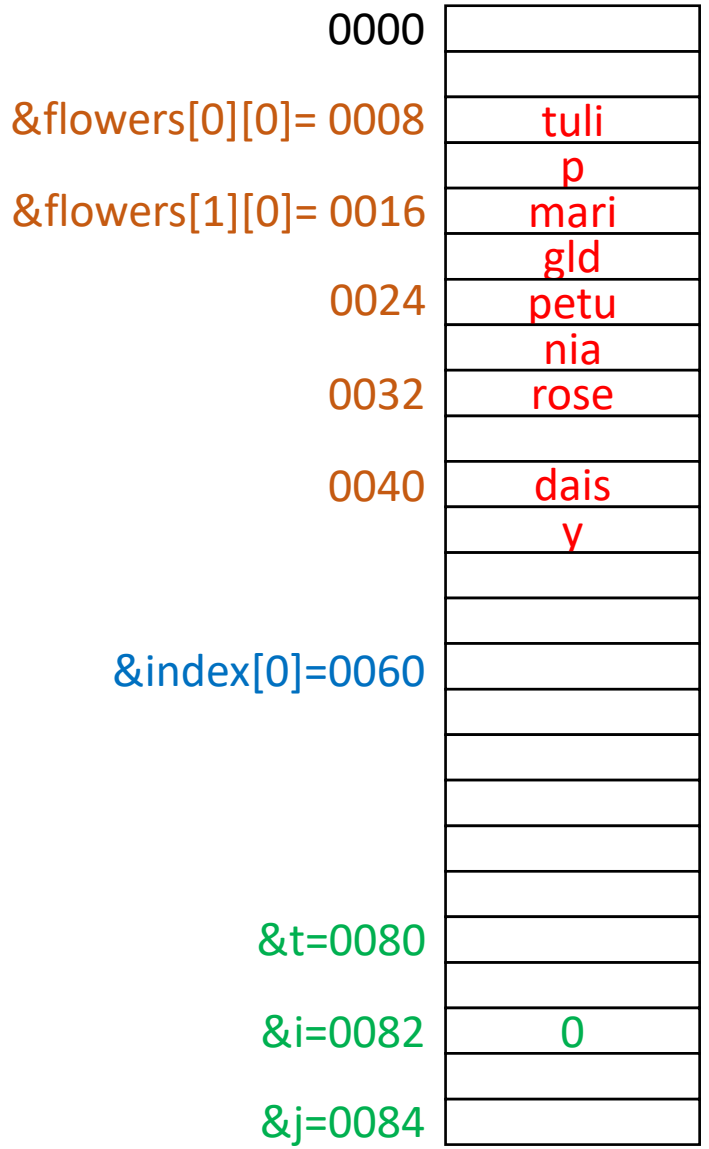
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Initialization of the address array pointing to the strings …

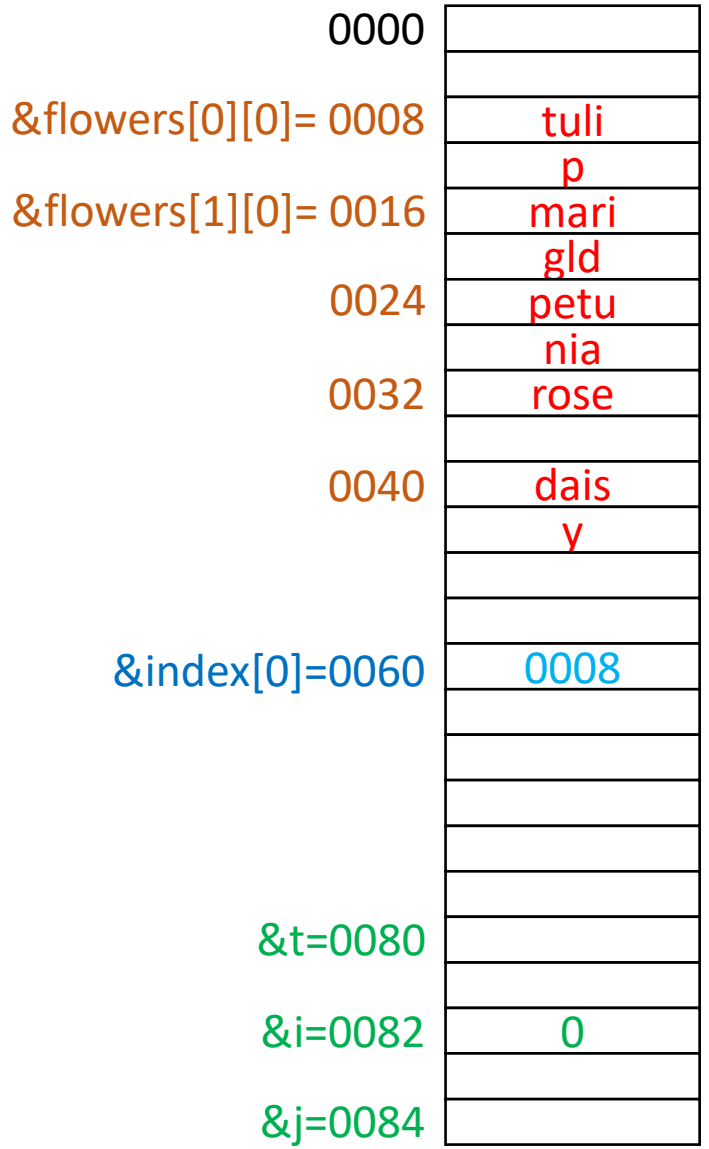
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Initialization of the address array pointing to the strings …

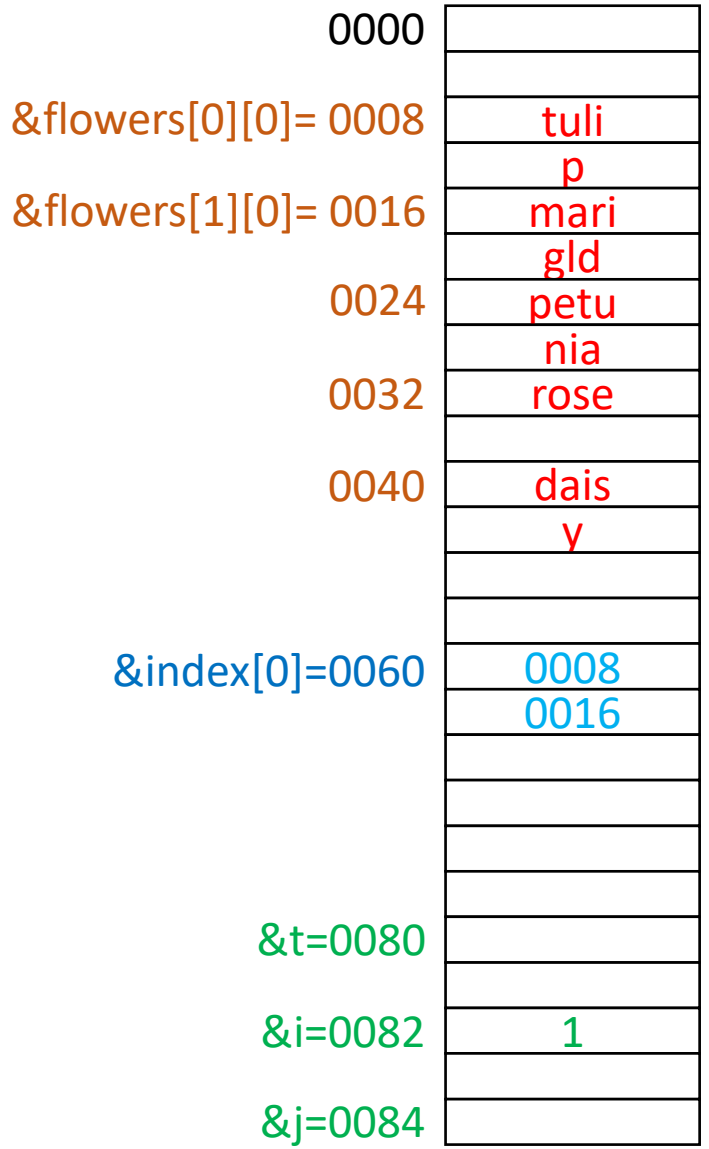
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Initialization of the address array pointing to the strings …

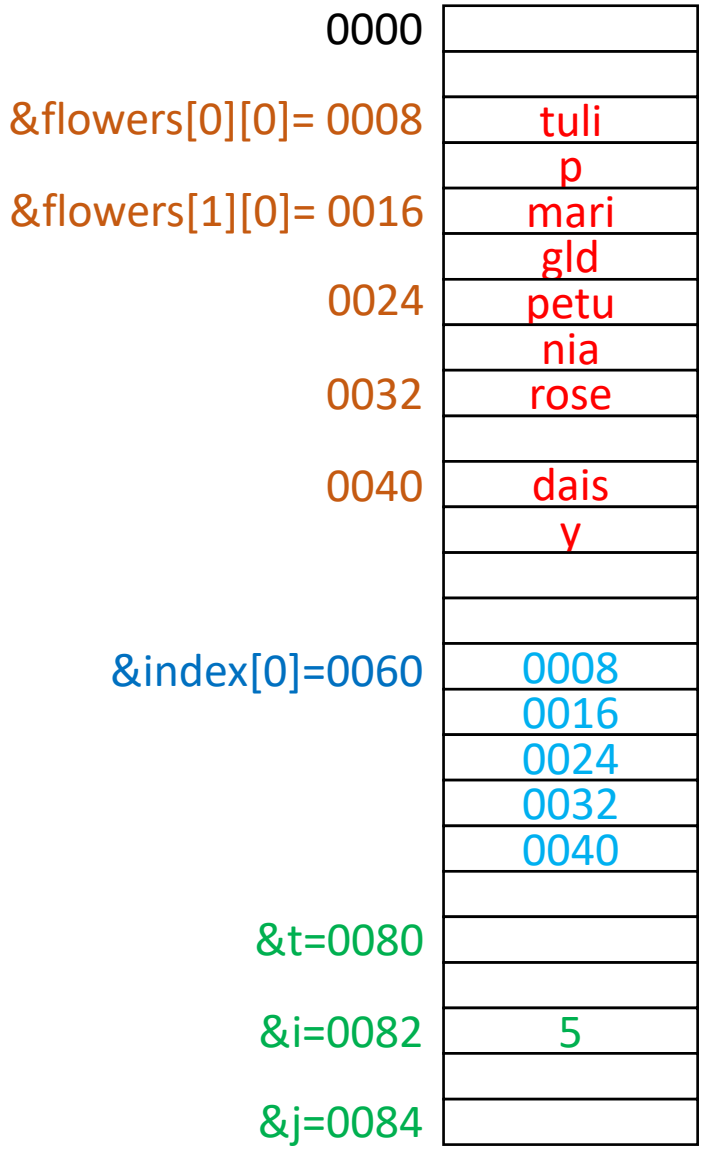
Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Se  addresses in the array to point to actual strings */
    for  ; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Array initialization finished…

Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Now printing the string using the array of addresses…

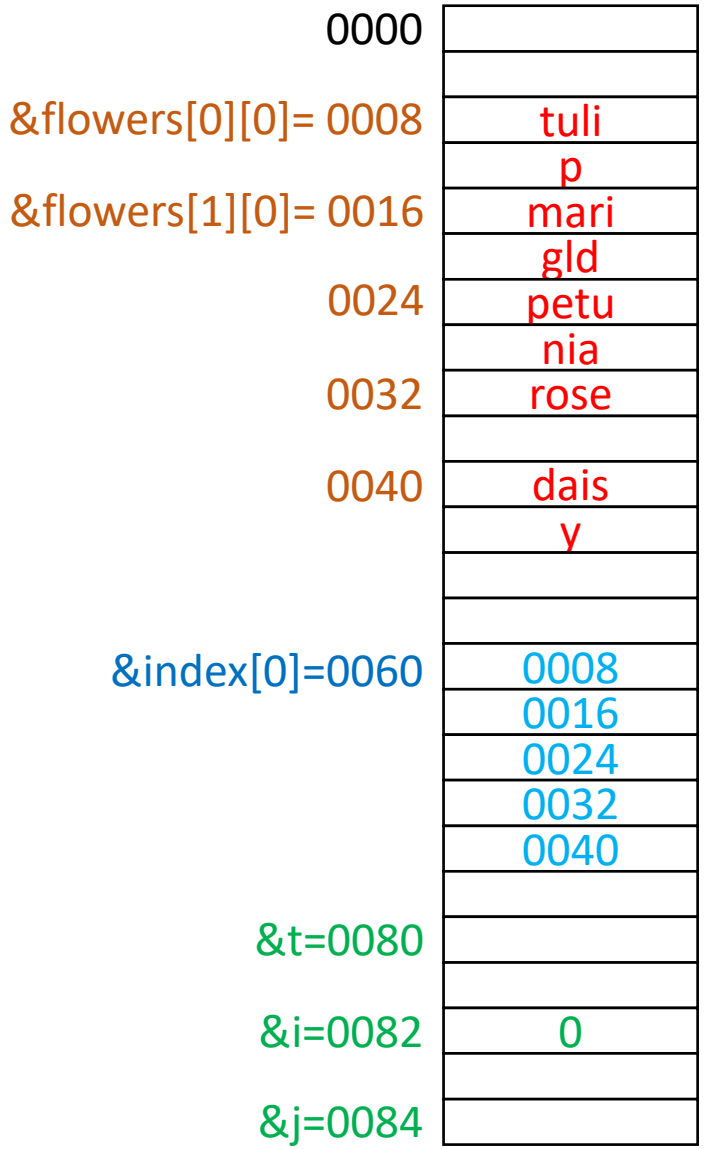
Abstract Memory Managed by Runtime (Compiler)

0000	
&flowers[0][0]= 0008	tuli
	p
&flowers[1][0]= 0016	mari
	gld
0024	petu
	nia
0032	rose
0040	dais
	y
&index[0]=0060	0008
	0016
	0024
	0032
	0040
&t=0080	
&i=0082	0
&j=0084	

```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<NUMFLOWERS; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Starting the sorting…

Abstract Memory Managed by Runtime (Compiler)



```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOW ; i++) index[i] = flowers[i];
    for (i=0; i<n;  printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NU WERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Second loop…

Abstract Memory Managed by Runtime (Compiler)

0000	
&flowers[0][0]= 0008	tuli
	p
&flowers[1][0]= 0016	mari
	gld
0024	petu
	nia
0032	rose
0040	dais
	y
&index[0]=0060	0008
	0016
	0024
	0032
	0040
&t=0080	
&i=0082	0
&j=0084	1

```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

First part of the swap…

Abstract Memory Managed by Runtime (Compiler)

0000	
&flowers[0][0]= 0008	tuli
	p
&flowers[1][0]= 0016	mari
	gld
0024	petu
	nia
0032	rose
0040	dais
	y
&index[0]=0060	0008
	0016
	0024
	0032
	0040
&t=0080	0008
&i=0082	0
&j=0084	1

```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Middle part of the swap…

Abstract Memory Managed by Runtime (Compiler)

0000	
&flowers[0][0]= 0008	tuli
	p
&flowers[1][0]= 0016	mari
	gld
0024	petu
	nia
0032	rose
0040	dais
	y
&index[0]=0060	0016
	0016
	0024
	0032
	0040
&t=0080	0008
&i=0082	0
&j=0084	1

```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

Last part of the swap…

Abstract Memory Managed by Runtime (Compiler)

0000	
&flowers[0][0]= 0008	tuli
	p
&flowers[1][0]= 0016	mari
	gld
0024	petu
	nia
0032	rose
0040	dais
	y
&index[0]=0060	0016
	0008
	0024
	0032
	0040
&t=0080	0008
&i=0082	0
&j=0084	1

```c
#define NAMELENGTH 8
#define NUMFLOWERS 5
void test() {
    char flowers[NUMFLOWERS][NAMELENGTH] =
        {"tulip", "marigld", "petunia", "rose", "daisy"};
    char * index[NUMFLOWERS];
    char * t;
    int i, j;
    /* Set up addresses in the array to point to actual strings */
    for (i=0; i<NUMFLOWERS; i++) index[i] = flowers[i];
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
    for (i=0; i<NUMFLOWERS; i++) {
        for (j=i+1; j<NUMFLOWERS; j++) {
            if (strcmp(index[i], index[j]) > 0) {
                t = index[i];
                index[i] = index[j];
                index[j] = t;
            }
        }
    }
    for (i=0; i<n; i++) printf("  %X: %s\n", index[i], index[i]);
}
```

After all iterations…

Abstract Memory Managed by Runtime (Compiler)

0000	
&flowers[0][0]= 0008	tuli
	p
&flowers[1][0]= 0016	mari
	gld
0024	petu
	nia
0032	rose
0040	dais
	y
&index[0]=0060	0040
	0016
	0024
	0032
	0008
&t=0080	0008
&i=0082	0
&j=0084	1