C PROGRAMING

Ketan Kore

Sunbeam Infotech

String

- String is character array terminated with '\0' character.
  - '\0' is character with ASCII value = 0.
- Example :

```
char arr[5] = "abcde";
int j;
for(j=0; j<5; j++)
        printf("%c",arr[ j ]);
```

- String input/output
  - char str[20];
  - scanf(“%s”,str); /*Input*/
  - printf(“%s”,str); /*Output*/
  - gets(str); /*Input*/
  - puts(str); /*Output*/
  - scanf("%[^\n]", str); // scan whole line

String functions

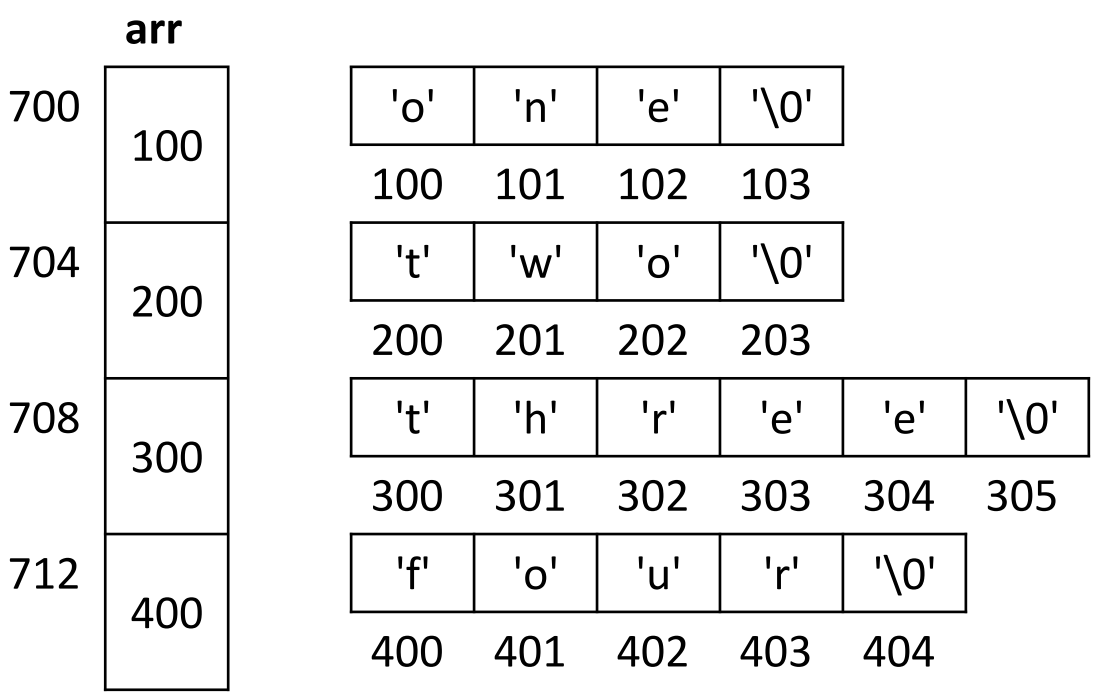
- C library have many string functions.
- They are declared in string.h
    - strlen() – size_t strlen(const char *s);
    - strcpy() – char* strcpy(char *dest, const char *src);
    - strcat() – char* strcat(char *dest, const char *src);
    - strcmp() – int strcmp(const char *s1, const char *s2);
    - strcmpi() - int strcmpi(const char *s1, const char *s2);
    - strchr() – char* strchr(const char *s, int ch);
    - strstr() – char* strstr(const char *s1, const char *s2);
    - strrev() – char* strrev(char *s);

NULL pointer

- If pointer is uninitialized, it will hold garbage address (local pointer variables).
- Accessing such pointer may produce unexpected results. Such pointers are sometimes referred as wild pointers.
- C defined a symbolic const NULL, that expands to (void*)0.
- It is good practice to keep well known address in pointer (instead of garbage).
- NULL is typically used to initialize pointer and/or assign once pointer is no more in use.
- Many C functions return NULL to represent failure.
    - strchr(), strstr(), malloc(), fopen(), etc.

Array of pointers

```
char *arr[] = { "one", "two", "three", "four" };
for(i = 0; i < 4; i++)
        puts(arr[i]);
```

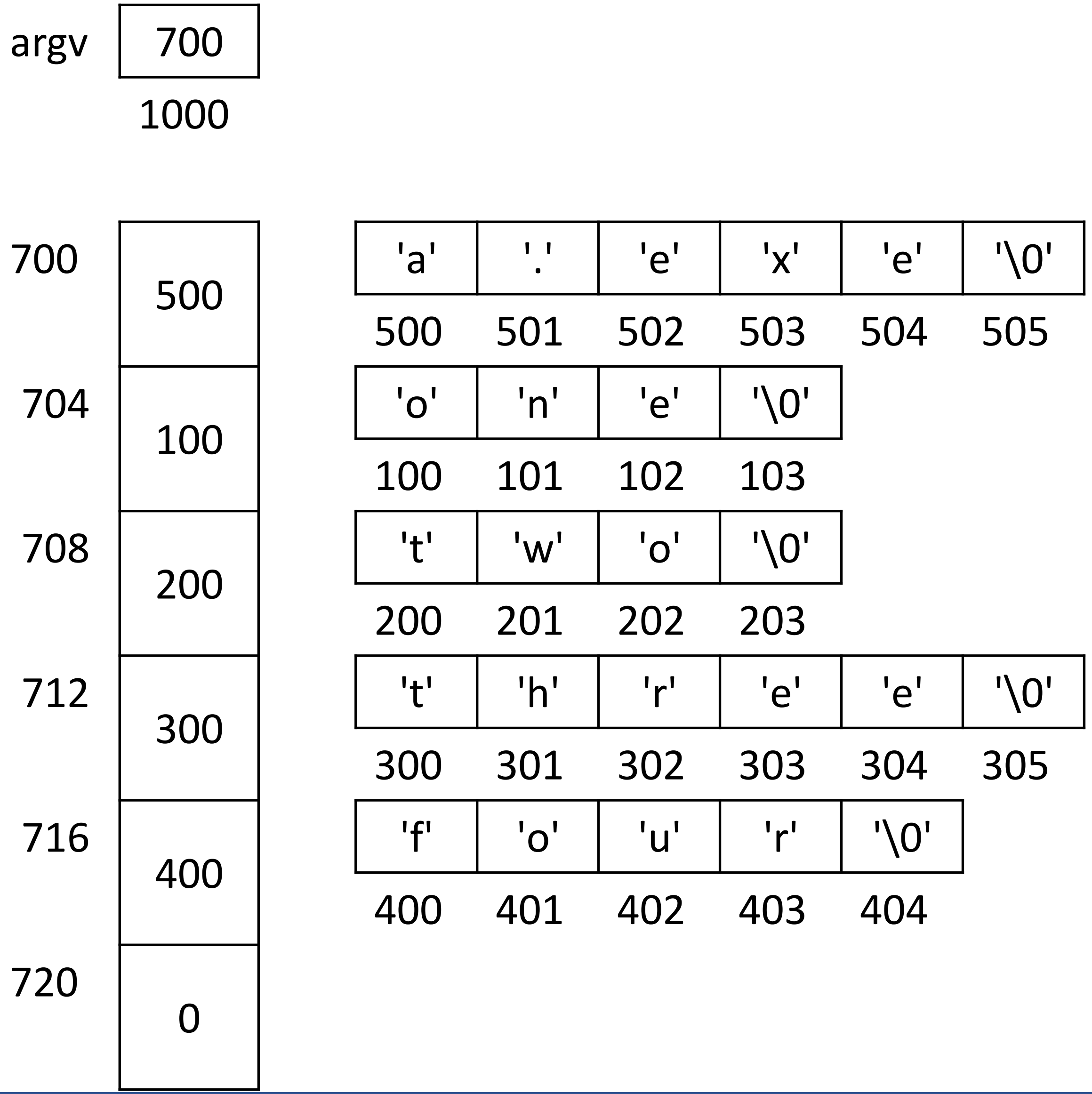


Command line arguments

- Command line arguments are information passed to the program while executing it on command line.
- cmd> a.exe one two three four

a.exe is also considered as one argument

```
int main(int argc, char *argv[]) {
        int i;
        for(i=0; i < argc; i++)
                puts(argv[i]);
        return 0;
}
```



void pointer

- Void pointer is generic pointer it can hold address of any data type (without casting).
- Scale factor of void* is not defined, so cannot perform pointer arithmetic.
- To retrieve value of the variable need type-casting.
- void* is used to implement generic algorithms.

2-D array

- Logically 2-D array represents m x n matrix i.e. m rows and n columns.
  - int arr[3][4] = { {1, 2, 3, 4}, {10, 20, 30, 40}, {11, 22, 33, 44} };

- Array declaration:
  - int arr[3][4] = { {1, 2, 3, 4}, {10, 20, 30, 40}, {11, 22, 33, 44} };
  - int arr[3][4] = { {1, 2 }, {10}, {11, 22, 33 } };
  - int arr[3][4] = { 1, 2, 10, 11, 22, 33 };
  - int arr[ ][4] = { 1, 2, 10, 11, 22, 33 };

	0	1	2	3
0	**1**	**2**	**3**	**4**
1	**10**	**20**	**30**	**40**
2	**11**	**22**	**33**	**44**

SUNBEAM

Thank you!

Ketan Kore <ketan.kore@sunbeaminfo.com>