תנאי ולולאות

if (condition)
{
statement 1;
statement 2;
}
else
{
statement 3;
statements 4;
}
	switch (variable)
	{
	case 1 : statements;
	break;
	case 2 : statements;
	break;
	case 3 : statements
	break;
	default : statements;
	}
		האופרטור הטרינארי
		(something) **?** ifTrue **:** ifFalse;
		
		if(num) equivalent to **if(num!=0)**
		
		if(!num) equivalent to **if(num==0)**

while (condition)
{
statements;
}
	do
	{
	statements;
	} while (condition);
		for (initialization; condition; increment)
		{
		statements;
		}
		
		* equivalent while loop *\

פעולות חשבוניות בסיסיות:

חיבור	`x = a + b;`
חיסור	`x = b - a;`
כפל	`x = a * b;`
חילוק	`x = b / a;`
שארית בחילוק (מודולו)	`x = c % a;`

מחרוזות

העתקת מחרוזת השניה אל הראשונה. ערך המוחזר – מחרוזת הראשונה.

char *strcpy (char *destination, char *source);

שירשור מחרוזת השניה אל סוף מחרוזת הראשונה .מתקבלת מחרוזת אחת שגם מוחזרת.

char *strcat (char *destination, char *source);

השוואת המחרוזות בסדר לקסיקוגרפי.מחזירים 0 אם זהות, אחרת – מספר חיובי (הראשונה "גדולה" מהשנייה או שלילי (הראשונה "קטנה" מהשנייה).

int strcmp (char *string1, char *string2);

חיפוש מופע ראשון של string2 בתוך string1.מחזירים מצביע לתחילת מופע או NULL

char *strstr (char *string 1, char *string2);

הפקודה מחזירה כמות התוים במחרוזת str בפועל, ללא תו של סיום המחרוזת.

unsigned int strlen(char *str)

המרות ממחרוזת למספרים ובחזרה

int n = atoi(char *str);
float n = atof(char *str);

char* itoa(int temp, char *str, int radix)

פקודה itoa ממירה מספר int הנמצא במשתנה temp למחרוזת שנשמרת בתוך str.
ערך המוחזר – אותה המחרוזת str. ניתן להשתמש בפקודה itoa ללא ערך המוחזר כלל.

קבצים

- פתיחת קובץ **FILE∗ fopen(char∗ filename , char∗ mode)**
- סגירת קובץ **fclose(FILE∗)**
- פונקציה לקריאת תו בודד מקובץ **int fgetc(ptr_to_file)**
- פונקציה לכתיבת תו בודד לקובץ **int fputc(ch, ptr_to_file);**
- קריאת מחרוזת מקובץ **char * fgets(string, n, ptr_to_file);**

הפקודה עוצרת כאשר מסיימת לקרוא n-1 תוים, או כאשר מגיעה ל-enter או כאשר מגיעה ל-EOF, מה שבא קודם. אם הצליחה לקרוא, מחזירה את string, אחרת NULL.

- פונקציה לכתיבת מחרוזת לקובץ **fputs(string, ptr_to_file);**
- **fprintf(ptr_to_file, print_format, value, value, value,....);**
- **fscanf(ptr_to_file, print_format, addrs1, addrs2, addrs3..);**
- הזזת מצביע בקובץ **void fseek (FILE∗ fp, long offset, int origin)**

הפרמטרים ל- origin הם:

0 (או SEEK_SET) - הזזה מתחילת הקובץ,

1 (או SEEK_CUR) – הזזה מהמקום הנוכחי,

2 (או SEEK_END) - הזזה מסוף הקובץ.

- **long ftell(FILE* f)**
- הפונקציה מחזירה את מספר הבתים מתחילת הקובץ ועד המקום הנוכחי של המצביע f

דוגמא להפעלת פקודה fopen :

fin = fopen(″example.txt″ ,″wt″);

דוגמא להפעלת פקודה fprintf :

fprintf(out, ″The number is %d″, num);

פרמטרים לפתיחת קובץ:

r	**Open a text file for reading**
w	**Create a text file for writing**
a	**Append to a text file**
r+	**Open a text file for read/write**
w+	**Create a text file for read/write**

הקצאה דינאמית

- **void* malloc(size_t size)**

temp=(info*)malloc(size*sizeof(info));

- **void* realloc(void *block , size_t size)**

temp=(info*)realloc(temp ,size*sizeof(info));

- **void* calloc(size_t nitems , size_t size)**

temp=(info*)calloc(size,sizeof(info));

- **free(void *p)**

free(temp);

פעולות על ביטים:

and	&
or	|
xor	^
not	~
shift left	<<
shift right	>>

a	b	a&b	a|b	a^b	~a
0	0	0	0	0	1
1	0	0	1	1	0
0	1	0	1	1	1

<u>מבנה של צומת ברשימה מקושרת חד-כיוונית</u>

```
typedef struct Node
 {          info data;
            struct Node *next;
} node,*p_node;
```

<u>מבנה של צומת ברשימה מקושרת דו כיוונית</u>

```
typedef struct Node
 {          info data;
            struct Node *prev;
            struct Node *next;
} node,*p_node;
```