Managing input and output operations

Reading a character

* reading a single charater can be done using the function getchar() and also scanf()

ch = getchar();

i/o statements - to perform basic i/o functions c provide library funtcions called stdio.h

ex: scanf(), printf(), getchar(), putchar(), getch(), gets(), puts() etc

Formatted i/o

Unformatted i/o

**1) Formatted I/O Statements** - this enables the user to specify the type of data and the way in which it should be read or written out

EX : scanf(), printf()

**2) Unformatted I/O Statements** - this do not specify the type of data and the way in which it should be read or written out

EX : getchar(), putchar(), getch(), gets(), puts()

**Scanf():-**

Syntax

scanf("Control String", address\_list);

where,

"control string" is a sequence of one or more character groups. (specifies the type of values)

"address list" are address of memory locations where the values of input variable should be stored.

%c - single character

%d - decimal integer

%f - floating point value

%u - unsigned value

**Integer input**

Ex : num = 386;

scanf("%3d",&num);

where, 3 is the field width of input number, other 4 and 5 are in temporary buffer wherein th erest of the place will be filled with 0.

* No space or any other character is used after te last character group.

Ex: scanf("%d%d%d ",p,q,r)

………………………………………………………………………………………………………………………

#include <stdio.h>

int main()

{

int i;

float f;

char ch;

char str1[20];

double d;

printf("\nEnter the proper values\n");

printf("\nEnter integer value: \n");

scanf("%d",&i);

printf("\nInteger value: \n");

printf("%d", i);

return 0;

}

printf - used for deferencing the value

* Checking in different ways

#include <stdio.h>

int main()

{

int i;

float f;

char ch;

char str1[20];

double d;

printf("\nEnter the proper values\n");

printf("\nEnter integer value: ");

scanf("%3d",&i);

printf("\nInteger value: %05d\n",i);

return 0;

}

if taken as above

input - 12345

output - 00123 (4 and 5 are in buffer)

* While scanning the string we should not specify the address of operator

ASCII value of enter key is 10

* For single character value - it checks in the temporary buffer first then if nothing available then comes to give user input.

To avoid this - fflush stdin( it is not working in linux environment)

So we can avoid by clearing of the buffer ie we need to add scanf(" ") where a space is used

it does not work for float because it only works for single character

Assignment : how can I avoid numeric constants

just a recap to push the code

git add.

git commit -m "updated"

git push -u origin main

Scan the employee records and print them in the particular format

name id gender address phone number salary and designation

print in the format

SlNo | ID | NAME | GENDER | ADDRESS | PHNO | SALARY | DESG

001 |101 |tejaswi|N |ch |333333|93288976| JD

note - when menu based items are used it is bettre to use getc rather than scanf

gets -- can assign and store it in the buffer

fgets should be specified with how many char and from where to be read-- it also stores one extra line

These both will read until the new line

--scanf reads until blank space and new line...to overcome this we use pattern matching

**DECISION MAKING AND BRANCHING**

--to test multiple conditions

1.if

2. if else

3. nested if else

4. switch statement

if(cond)

statement

#include <stdio.h>

int main()

{

int age;

scanf("%d",&age);

if(age>=18)

{

printf("\nYou are eligible to vote");

}

printf("\nIndian Citizen\n");

return 0;

}

**----nested if else**

if(cond1)

{

if(cond2)

{

}

else

{

}

}

else

{

if(cond3)

{

}

else

{

}

}