**A SIMPLE CALCULTOR WITH GRAPHICS**

A Assembly language Report Submitted by:

1. *DANELLA PATRICK(CSC-18F-169)*

Students of

BACHELOR OF SCIENCE

IN

COMPUTER SCIENCE

at



**SINDH MADRESSATUL ISLAM UNIVERSITY**

**KARACHI (PAKISTAN)**

**Monday, 28th August , 2020**

**TABLE OF CONTENT**

1. **INTRODUCTION**
   1. Purpose
   2. Definition, Acronyms and Abbrevation
   3. Intented Audience
   4. Project Scope
   5. Overview
2. **OVERALL DESCRIPTION:**
   1. Product Prespective
   2. Product Features
   3. Product Functions
   4. Constarints
      1. *Hardware*
      2. *Software*
3. **SPECIFIC REQUIREMENT**
   1. Functional Requirement
   2. Hardware Specification
   3. Software Specification
      1. Tasm
   4. System Analysis
      1. Flowchart
   5. Interface
      1. User Interface
4. **INTRODUCTION:** 
   1. **Purpose:**

I have tried my best to make process of Calculator as simple as possible using the structure of Assembly language. I have tried to design the Project in such a way that user may not have any difficulty in using our program. Even though we cannot claim that this work to be entirely exhaustive the main purpose of our exercise is to perform each employee’s activity in computerized way rather than manually which is time consuming.

* 1. **Definition, Acronyms and Abbrevation:**

Register:

bl: base register

al: accumaltor register

ax: base regiter (16 bits)

bx: base resgiter (16 bits)

mov : Is use to set positon

int 10h: Graphics

int 21h :

01: input

02:output

Lea; load effective register

* 1. Intented Audience

The project is for the students and the teachers who use calculators.

* 1. **Project Scope:**

Simple Calculator is a program based on assembly language.It is a simple program which takes imput from the user and calculate the input according to the operator selected by the user .

* 1. **Overview:**

This report is organized into two parts the first is the overall description and the second is the specific requirement .The Overall description will describe the Hotel Management System.The specific requirement section describes the detail of the system.

1. **OVERALL DESCRIPTION:**
   1. **Product Prespective:**

The Project simple calcultor is a type of a system that makes easier for the user to perform calculation . My Project is more convenient and user-friendly.

* 1. **Product Features:**
* Graphical Inteface
* User can input number acoording to his/ her wish.
* Addition
* Multiplication
* Divison
  1. **Product Functions :**

|  |  |
| --- | --- |
| Function | Make Reservation |
| Input | Numbers , operand |
| Output | A answer is showm |

This simple calcultor will take input from the user and will ask for the operation to perform on the inputs it will then give the diserable output.

* 1. **Constraints:**
     1. ***Hardware:***
* Operating System: Supports operating system such as Window.
* Computer : 512MB + RAM , monitor with minimum resolution.
* Hard Drive: should be in NFTS file-system formatted with minimum 10GB free space
  + 1. ***Software:***
* Software is designed to run on any Microsoft Window 7 (32 bits)
* Tasm or any other assembly language editor
* Notepad or Notepadd ++ or GUI assembler

1. **SPECIFIC REQUIREMENT:** 
   1. **Functional Requirement:**

* Make Calcution
* Take Inputs
* User Friendly
  1. **Hardware Specification**
* 2-80GB hard disks
* Computer memory of >1.5GB and processor speed of 3.0GHz
* Windows vista operating system
* Input devices: Keyboards, Mouse, An OMR 29 ABC
* Output devices: Flat panel monitors, speakers, Printers
  1. **Software Specification:**

**3.3.1. *Tasm:*** Turbo Assembler (TASM) is a computer assembler (software for program development) developed by Borland which runs on and produces code for 16- or 32-bit x86 DOS or Microsoft Windows. It can be used with Borland's high-level language compilers, such as Turbo Pascal, Turbo Basic, Turbo C and Turbo C++. The Turbo Assembler package is bundled with the Turbo Linker, and is interoperable with the Turbo Debugger. TASM can assemble Microsoft Macro Assembler (MASM) source using its MASM mode and has an ideal mode with a few enhancements. Object-Oriented programming has been supported since version 3.0. The last version of Turbo Assembler is 5.4, with files dated 1996 and patches up to 2010; it is still supplied with Delphi and C++Builder.

TASM itself is a 16-bit program; it will run on 16- and 32-bit versions of Windows, and produce code for the same versions. There are ways to run 16-bit programs such as TASM on 64-bit Windows (e.g., on a virtual machine), but it will not generate 64-bit Windows code. **System Anaylsis:**

**3.6.1.** **Flowchart:**

INPUT

NUMBER 1, NUMBER2

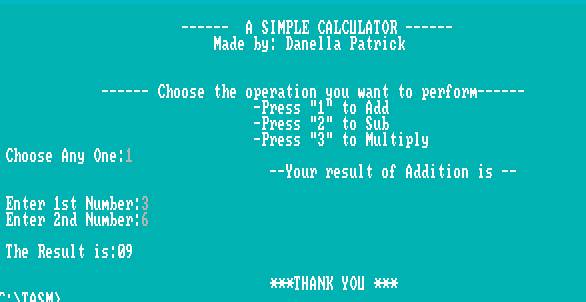
OUTPUT

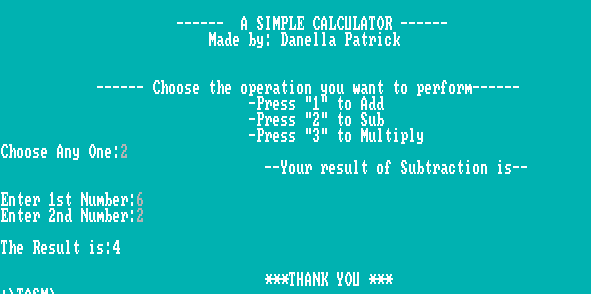
process

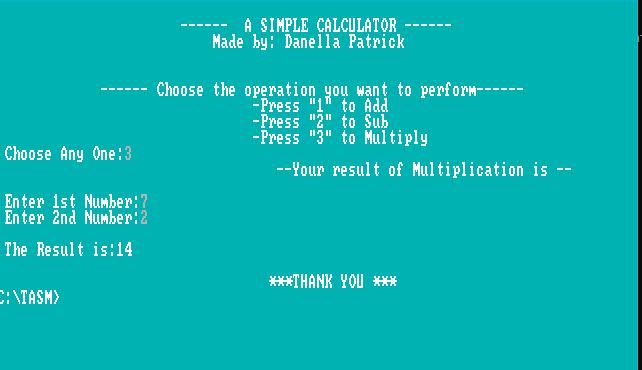
ASSIGN OPERRAND

* 1. **Interface:**

**3.5.1.User Interface:**







**3.5.2. Code:**

.model large

.stack 100h

.data

char db " "

message1 db 0ah,0dh,' ------ a simple calculator ------ ','$'

message2 db 0ah,0dh,' made by: danella patrick ','$'

space1 db 0ah,0dh,'','$'

space2 db 0ah,0dh,'','$'

message3 db 0ah,0dh,' ------ choose the operation you want to perform------ ','$'

message4 db 0ah,0dh,' -press "1" to add ','$'

message5 db 0ah,0dh,' -press "2" to sub ','$'

message6 db 0ah,0dh,' -press "3" to multiply ','$'

message8 db 10,13,' choose any one:$'

input1 db 10,13,10,13,' enter 1st number:$'

input2 db 10,13,' enter 2nd number:$'

res db 10,13,10,13,' the result is:$'

thanks db 10,13,10,13,' \*\*\*thank you \*\*\*$'

num1 db ?

num2 db ?

result db ?

.code

start:

mov ah,0

mov al,14

int 10h

mov ah,0bh

mov bh,00h

mov bl,03

int 10h

mov bh,1

mov bl,1

int 10h

mov ax,@data

mov ds,ax

lea dx, message1

mov ah,09

int 21h

lea dx, message2

mov ah,09

int 21h

lea dx, space1

mov ah,09

int 21h

lea dx, space2

mov ah,09

int 21h

lea dx, message3

mov ah,09

int 21h

lea dx, message4

mov ah,09

int 21h

lea dx, message5

mov ah,09

int 21h

lea dx, message6

mov ah,09

int 21h

lea dx, message8

mov ah,09

int 21h

mov ah,01

int 21h

mov bh,al

sub bh,48

cmp bh,1

je add

cmp bh,2

je sub

cmp bh,3

je mul

add:

lea dx,input1 ;enter 1st number

mov ah,9

int 21h

mov ah,1

int 21h

mov bl,al

lea dx,input2 ;enter 2nd number

mov ah,9

int 21h

mov ah,1

int 21h

mov cl,al

add al,bl

mov ah,0

aaa

mov bx,ax

add bh,48

add bl,48

lea dx,res

mov ah,9

int 21h

mov ah,2

mov dl,bh

int 21h

mov ah,2

mov dl,bl

int 21h

jmp exit\_p

sub:

lea dx,input1 ;enter 1st number

mov ah,9

int 21h

mov ah,1

int 21h

mov bl,al

lea dx,input2 ;enter 2nd number

mov ah,9

int 21h

mov ah,1

int 21h

mov cl,al

sub bl,cl

add bl,48

lea dx,res

mov ah,9

int 21h

mov ah,2

mov dl,bl

int 21h

jmp exit\_p

mul:

lea dx,input1

mov ah,9

int 21h

mov ah,1

int 21h

sub al,30h

mov num1,al

lea dx,input2

mov ah,9

int 21h

mov ah,1

int 21h

sub al,30h

mov num2,al

mul num1

mov result,al

aam

add ah,30h

add al,30h

mov bx,ax

lea dx,res

mov ah,9

int 21h

mov ah,2

mov dl,bh

int 21h

mov ah,2

mov dl,bl

int 21h

jmp exit\_p

exit\_p

lea dx,thanks

mov ah,9

exit:

mov ah,4ch

int 21h

end start