Laborator 1+2

Lectii:

- Lectia 1 (Tipuri de date, registrii, instructiuni de baza)
- Lectia 2 (Salturi, comparatii, instructiuni de control si bucle)

Stuff:

- Cu pointeri si tablouri vom lucra umpic mai tarziu.

Exercitii rezolvate:

1. Implementati o functie care returneaza maximul a doua numere.

```
// Lab9.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include <iostream>
using namespace std;

int max(int a, int b) {
 int maxim;
 _asm {
  //completati
  mov eax, a //in eax punem valoarea variabilei a, pentru ca instructiunea cmp suport
  cmp eax, b //comparam eax (care are stocata valoarea lui a), cu b. Instructiunea cm
  jle b_is_max // JLE - Jump if lower or equal : Se va sari la label b_is_max daca a
  mov maxim, eax //Daca suntem aici in executie => comparatia de mai sus nu a dus la
        //Deci a > b => in variabila "maxim" punem valoarea lui a, memorata in eax.
  jmp end_if //Sarim la label-ul end_if, neconditional.
 b_is_max:
  mov eax, b  //Daca am ajuns aici inseamna ca b este maxim. Nu putem scrie "mov maxi
     //Folosim registrul eax ca ajutor.
  mov maxim, eax
 end_if:
 }
 //Variabila maxim va avea valoarea corecta.
 return maxim;
}

int main()
{
 int a, b;
 cout<< "a = ";
 cin >> a;

 cout << "b = ";
 cin >> b;
 printf("MAX(a,b) = %d", max(a, b));
}
```

2. Implementati o functie care verifica daca un numar este palindrom. Se va returna 0 daca nu, si 1 daca da.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
```

```
/*
 int rev = 0;
 int nn = n;
 while(n > 0){
  rev = rev* 10 + n%10;
  n/=10;
 }
 if(nn == rev){
  return 1;
 }
 return 0;
*/
int palindrom(int n) {
 int retVal = 0;
 int zece = 10; //Mul si div suporta ca operanzi doar registrii sau adrese de memorie

 //Pe viitor vom vedea cum putem folosi memoria pentru a salva valoarea unui registru
 _asm {
  //Completati
  mov ebx, n //In ebx vom patra valorea lui n pe care o vom imparti repetat la 10.
  mov ecx, 0 //In ecx vom calcula numarul inversat.
 start_while:
  cmp ebx, 0
  je end_while
  //Inmultim ce avem calculat pana acum cu 10. (rev = rev * 10)
  mov eax, ecx
  xor edx, edx
  mul zece
  mov ecx, eax
  //Impartim numarul la 10. Restul il vom aduna la numarul inversat, iar catul il vom
  mov eax, ebx
  xor edx, edx // Pentru ca lucram cu registrii pe 32 bits, impartirea va fi de fapt
  div zece
  mov ebx, eax //Catul. n = n / 10;
  add ecx, edx //Restul. // rev += n % 10;
  jmp start_while
 end_while:
  cmp n, ecx
  jne diferite
  mov retVal, 1
  jmp end_if
 diferite:
  mov retVal, 0
 end_if:
 }
 return retVal;
}

int main()
{
 int a;
 cout << "a= ";
 cin >> a;
 int retVal = palindrom(a);
 cout << retVal << '\n';
 if (retVal == 1) {
  cout << "Numarul este palindrom\n";
 } else {
  cout << "Numarul nu este palindrom\n";
 }
}
```

Exercitii nerezolvate:

1. Scrieti o functie care primeste ca parametru un numar **n** si returneaza suma 1 + 2 + ... + n, fara a folosi o formula.
2. Scrieti o functie care primeste ca parametru un numar **n** si returneaza al n-lea termen din sirul

Fibonacci. Nu folositi tablouri sau recursivitate.

3. Scrieti o functie care oglindeste bitii unui numar. Se va lucra fara semn (unsigned int). Exemplu : 140 -> 822083584

Comments

You do not have permission to add comments.