**BLGM224 Sayısal Mantık Sistemleri**

**MARIE Assembly Dili Programlama Örnek Sorular**

**Önemli Bilgiler:**

***I. Aşağıda verilen 5. ve 6. soruların cevaplarını hazırlayarak 20 MAYIS 2019 Pazartesi, saat 16:30’da CMPE227 nolu lab. sınıfında dersin asistanına gösteriniz.***

***II. Aşağıda verilen 8. ve 10. soruların cevaplarını hazırlayarak 27 MAYIS 2019 Pazartesi, saat 16:30’da CMPE227 nolu lab. sınıfında dersin asistanına gösteriniz.***

**S.1. Üretilme kuralı F0=0, F1=1, F2=3, Fi=Fi-1+Fi-2+Fi-3, i > 2, şeklinde verilen sayı dizisinin ilk 10 elemanını üreten Marie Assembly Dili Programını (MADP) yazınız.**

**S.2. Klavyeden okunan bir tam sayının çift veya tek olduğunu bulmak amacıyla *IkiyeBol* isimli bir foksiyonu içeren bir MADP yazınız.**

**S.3. z= (x\*y+3x+3y)/x\*y -2x -2y) ifadesini x=20 , y=10 için hesalayan bir MADP yazınız. Bölümü Q değişkeninde, kalanı ise R değişkeninde saklayınız.**

**S.4. A, B, C ve D değişkenlerinin değerlerini klavyeden okuyup A-B, A-B-C ve A-B-C-D işlemlerini tek bir CıkarmaXY fonksiyonu üzerinden gerçekleştiren bir MADP yazınız. CıkarmayXY fonksiyonu (X-Y) işlemi için hazırlanmalıdır.**

**S.5. (EXP-1) “BLGM224 U SEVIYORUM” dizesini çıkış alanına yazdıran bir MADP hazırlayınız**

**S.6. (EXP-1) A=[2,3,5,8,4,8,1,9,3] vectöründeki tamsayı elemanların toplamını bulan bir MADP yazınız.**

**S.7. A=[1,2,3,4,5], B=[6,7,8,9,10] olarak tanımlanan vektörlerin toplamını hesaplayan bir MADP yazınız. Sonucu C vektöründe saklayınız.**

**S.8. (EX-2) Verilen iki A ve B vektörlerindeki elemanların maksimumlarını hesaplayan bir MADP yazınız. Sonucu C vectöründe saklayınız. Ci=max(Ai,Bi) işlemini gerçekleştirecektir.**

**S.9. Verilen bir A vektöründeki elemanların mutlak değerlerini hesaplayan bir MADP yazınız.**

**S.10. (EXP-2) Verilen bir A vektöründeki elemanların ikili eşleniklerini (2’s complement) hesaplayan bir MADP yazınız.**

**MARIE KOMUT KÜMESİ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Komut Tipi** | **Komut** | **Hex Opcode** | **Özet** |
| Aritmetik | Add X | 3 | X adresindeki değeri AC’ye ekle, sonucu AC’de sakla:  AC, AC ← AC + M[X] |
| Subt X | 4 | X adresindeki değeri AC’den çıkar, sonucu AC’de  sakla: AC ← AC - X |
| AddI X | B | X adresindeki değeri adress olarak alıp, ulaşılan değeri  AC’ye ekle, sonucu AC’de sakla: AC🡨 AC+M[M[X]] |
| Clear | A | AC ← 0 |
| Veri Aktarma | Load X | 1 | X adresinin içeriğini AC’ye yükle: AC🡨M[X] |
| Store X | 2 | AC’nin içeriğini X adresine yükle: M[X]🡨AC |
| I/O | Input | 5 | Kullanıcının bir değer girmesini iste ve  AC🡨InREG |
| Output | 6 | AC’nin içeriğini OutREG’e yükle ve çıkış alanında  göster: OutREG 🡨 AC. |
| Karar Verme | Jump X | 9 | X adresine atla: PC 🡨X |
| Skipcond (C) | 8 | Bir sonraki komutu C koşuluna bağlı olarak atla.  C koşulu AC’nin içerğiyle belirlenir ve şöyle tanımlanır:  - 000: if AC < 0, sonraki komutu atla - 400: if AC = 0, sonraki komutu atla - 800: if AC > 0, sonraki komutu atla |
| Fonksiyon | JnS X | 0 | Atla ve Sakla: PC’nin içeriğini X adresinde saklar ve  ve PC’yi X+1 ile yükler: M[X] 🡨PC, PC 🡨 X+1 |
| JumpI X | C | PC’yi X adresinin içeriğiyle yükler;  PC🡨 M[X]. |
| Dolaylı Adresleme | StoreI X | D | AC’nin içeriğini X adresindeki adresin içeriği olarak  Yükler: M[M[x]] 🡨 AC. |
| LoadI X | E | X adresindeki adresin içeriğini AC’ye yükler:  AC🡨M[M[X]]. |
|  | Halt | 7 | End the program |