Zadatak 2. Napišite prijevod u asemblerski jezik sljedećeg C programa:

```
void IzjednaciZbroj(int &a,int &b);
/* obje unesene varijable (a i b) dobivaju vrijednost njihove sume*/
{
      int sum;
      sum = a + b;
   a = sum;
      b = sum;
}

void main()
{
      int a = 5, b = 4;
      IzjednaciZbroj(a,b);
   printf("%d, %d", a, b);
}
```

```
#define a1 DWORD(M_[esp+4]) //adresa varijable a
#define b1 DWORD(M_[esp+8]) //adresa varijable b


PROC(IZJEDNACI)

      MOV(eax, DWORD(M_[a1])) //spremamo vrijendost a u eax
      ADD(eax, DWORD(M_[b1])) //uvecavamo eax za b
      MOV(DWORD(M_[b1]), eax) //spremamo eax u a
      MOV(DWORD(M_[a1]), eax)    //spremamo eax u b
      RET(0)
      ENDP


#define a DWORD(M_[ebp-4]) //lokalna varijabla a
#define b DWORD(M_[ebp-8]) //lokalna varijabla b


PROC (MAIN)
      PUSH(ebp)
      MOV(ebp, esp)

      SUB(esp, 8) //rezerviramo u stogu za 2 DWORLD-a
      MOV(a, 5)     //spremamo a
      MOV(b, 4)     //spremamo b

      //argumenti funkcije (s desna na lijevo)
      PUSH(ADR(b)) //adresa varijable b
      PUSH(ADR(a)) //adresa varijable a

      CALL(IZJEDNACI) //poziv funkcije, odnosno procedure
      ADD(esp, 8) //uvecavamo stog kako bi ponistili argumente

      MOV(eax, a)
      PUTI(eax) //ispisivamo a
      MOV(eax, b)
      PUTS(", ")
      PUTI(eax) //ispisivamo b

      MOV(esp, ebp)
      POP(ebp)
      RET(0)
      ENDP
```

Zadatak . Napišite prijevod u asemblerski jezik sljedećeg C programa:

```
int Bigger(int a,int b);
/* funkcija vraća vrijednost većeg od argumenata a i b*/
{
        If(a>b) return a; else return b;
}

void main()
{
        int a = 5, b = 4;
    printf("%d", Bigger(a, b));
}
```

```
#define a1 DWORD(M_[esp+4]) //argument 1
#define b1 DWORD(M_[esp+8]) //argumewnt 2

PROC(BIGGER)
       MOV(eax, a1)
       MOV(ebx, b1)
       CMP(eax, ebx) //usporedujemo argumente
       JG(L1) //ako je veci prvi, skoci na L1
       MOV(eax, b1) //ako nije ubaci u eax drugi argument
       JMP(L2) //skoci na L2
L1:
       MOV(eax, a1) //ubaci u eax prvi argument
L2:

       RET(0)
       ENDP


#define a DWORD(M_[ebp-4]) //var a
#define b DWORD(M_[ebp-8]) // var b
PROC (MAIN)
       PUSH(ebp)
       MOV(ebp, esp)

       SUB(esp, 8) //rezerviramo stog za 2 DWORD-a

       //spremanje lokalnih varijabli
       MOV(a, 5)
       MOV(b, 4)

       //slanje argumenata
       PUSH(b)
       PUSH(a)


       CALL(BIGGER)
       ADD(esp, 8)

       PUTI(eax) //ispisivanje povratka funckije
       MOV(esp, ebp)
       POP(ebp)
       RET(0)
       ENDP
```

Zadatak: Napišite prijevod u asemblerski jezik sljedećeg C programa:

```
int SumaNiza(int N, int niz[]);

void main()
{
  int  x[4] = {77, 1, 22, 3};
  printf("Rezultat: %d", SumaNiza(4,x));
}

int SumaNiza(int N, niz[])
{
  int i,sum;
  for (i=0; i<N; i++)
     sum +=  niz[1];
  return sum;
}
```

```
//elementi niza
#define a DWORD(M_[ebp-4])
#define b DWORD(M_[ebp-8])
#define c DWORD(M_[ebp-12])
#define d DWORD(M_[ebp-16])

PROC_DECL(SUMANIZA)

PROC (MAIN)
       PUSH(ebp)
       MOV(ebp, esp)

       SUB(esp, 16) //niz lokalno spremljen, cetiri clana

       //spremamo ih u stog
       MOV(a, 77)
       MOV(b, 1)
       MOV(c, 22)
       MOV(d, 3)

       //za argumente prenosimo adresu prvog niza i broj 4
       PUSH(ADR(a))
       PUSH(4)

       CALL(SUMANIZA)
       ADD(esp, 16)

       PUTS("Rezultat: ")
       PUTI(eax)
       MOV(esp, ebp)
       POP(ebp)
       RET(0)
       ENDP


#define niz DWORD(M_[ebp+12])  //drugi argument, clan niza
#define N DWORD(M_[ebp+8]) //prvi argument, broj 4
#define sum DWORD(M_[ebp-4])



PROC(SUMANIZA)
       PUSH(ebp)
       MOV(ebp, esp)
```

```
	SUB(esp, 4) //za lokalnu varijablu
	MOV(sum, 0)
	MOV(edi, 0) //i
	MOV(ebx, niz) //adresa niza


petlja:
	CMP(edi, N) //provjeravamo jel i manji od N
	JL(L1) //skoci ako jest
	JMP(L2)

L1:
	MOV(eax, sum) //prebaci sumu u eax
	ADD(eax, DWORD(M_[ebx-edi*4])) //uvecaj sumu, idemo s nizom prema donjim adresama jer su tako
dodani clanovi u memoriji
	MOV(sum, eax) //vracamo novu vrijednost sume u sum
	ADD(edi, 1) //inkrementiramo i (moze i INC(edi))
	JMP(petlja)
L2:

	MOV(eax, sum) //vrijednost sume zapisujemo u eax

	MOV(esp, ebp)
	POP(ebp)
	RET(0)
	ENDP
```

Zadatak: Napišite prijevod u asemblerski jezik sljedećeg C programa:

```
void Increment(int &a);
/* vraca  a  referencu s inkrementiranom vrijednošću */
{
      a += 1;
}


void main()
{
      int a = 5;
    Increment(a);
    printf("a = %d", a);
}
```

```
#define a1 DWORD(M_[esp+4]) //argument - adresa od a

PROC(INCREMENT)
       MOV(eax, DWORD(M_[a1])) //dohvacamo vrijednost s adrese i spremamo u eax
       ADD(eax, 1) //uvecavamo eax za 1
       MOV(DWORD(M_[a1]), eax)    //na istu adresu rvacamo uvecanu vrijednost
       RET(0)
       ENDP


#define a DWORD(M_[ebp-4]) //za lokalnu varijablu a

PROC (MAIN)
       PUSH(ebp)
       MOV(ebp, esp)

       SUB(esp, 4) //rezerviramo za a
       MOV(a, 5) //spremamo a

       PUSH(ADR(a)) //spremamo adresu od a      kao argument
       CALL(INCREMENT) //poziv
       ADD(esp, 4)

       PUTS("a = ")
       PUTI(eax) //ispisujemo eax (sto je funkcija vratila)

       MOV(esp, ebp)
       POP(ebp)
       RET(0)
       ENDP
```

Zadatak: Napišite prijevod u ASMC asemblerski jezik sljedeće C funkcije:

```
int ProduktElemenataNiza(int niz[], int brojelemenata);
{
  int i=0;
  int prod = 1;
  while(i < brojelemenata)
      prod *= niz[i++];
  return prod;
}
```

```
PROC(PRODUKTELEMENATA)
	PUSH(ebp)
	MOV(ebp, esp)

	SUB(esp, 4) //treba jedna lokalna varijabla (produkt)
	MOV(prod, 1) //prod postavljamo na 1
	MOV(edi, 0) //edi po obicaju korisimo za kretanje nizom (dakle varijabla i)
	MOV(ebx, niz) //spremamo adresu niza u ebx


	//petljica :)
petlja:
	CMP(edi, N) //usporedujemo i s brojem clanova N
	JGE(izlaz)  //ako je i veci ili jednak broju clanova, petlja je zavrsila, kreni na izlaz


	//ako nije
	MOV(eax, DWORD(prod))  //spremamo vrijednost produkta u eax
	MUL(eax, DWORD(M_[ebx +edi*4]))  //zbrajamo s clanom niza
	MOV(prod, eax) //vrijednost produkta spremamo u prod
	INC(edi) //uvecavamo i
	JMP(petlja) //na pocetak petlje

izlaz:
	MOV(eax, prod) //funkcija vraca prod
	MOV(esp, ebp)
	POP(ebp)
	RET(0)
	ENDP
```

Zadatak 2. Napišite prijevod u ASMC asemblerski jezik sljedeće C funkcije, koja svaki element niza A1, od N elemenata, uveća s odgovarajućim elementom niza A2.

```
void UvecajNizNizom(int A1[], int A2[], int N)
{
  int i=0;
  while(i < N) {
      A1[i] += A2[i];
      i++;
  }
}
```

```
/nema lokalnih u metodu pa krecemo od 4 i koristimo esp
#define A1 DWORD(M_[esp+4]) //prvi argument (niz)
#define A2 DWORD(M_[esp+8]) //drugi argument (niz)
#define N DWORD(M_[esp+12]) //treci argument (N)

PROC(UvecajNizNizom)

	MOV(edi, 0) //i=0
	MOV(ebx, A1)
	MOV(ecx, A2)


	//petljica
petlja:
	CMP(edi, N) //usporedujemo i s brojem clanova N
	JGE(izlaz)  //ako je i veci ili jednak broju clanova, petlja je zavrsila, kreni na izlaz

	//ako nije
	MOV(eax, DWORD(M_[ebx +edi*4]))  //spremamo vrijednost i-tog clana prvog niza u eax
	ADD(eax, DWORD(M_[ecx +edi*4]))  //dodajemo na to vrijednost i-tog clana drugog niza u eax
	MOV(DWORD(M_[ebx +edi*4]), eax) //vrijednost spremamo u i-ti clana prvog niza
	PUTI(eax)
	PUTS(" ")

	INC(edi) //uvecavamo i
	JMP(petlja) //na pocetak petlje

izlaz:
	RET(0)
	ENDP
```

Zadatak 2. Napišite prijevod u ASMC asemblerski jezik sljedećeg C programa, koji ispisuje sumu od N elemenata globalnog niza A.

```
int A[] = {1, 2, 3 , 4, 5};
int N = 5;

void main()
{
  int sum = 0, i = 0;
  while(i < N) {
      sum += A[i];
      i++;
  }
  printf("%d", sum);
}
```

```
VAR niz DD(5) = {D_(1), D_(2), D_(3), D_(4), D_(5)};
VAR N DD(1) = {D_(5)};

#define sum DWORD(M_[esp-4])


PROC (MAIN)
       PUSH(ebp)
       MOV(ebp, esp)
       SUB(esp, 4) //rezerviramo za sum

       MOV(edi, 0) //i=0
       MOV(sum, 0)
       MOV(ebx, DWORD(N)) //bez obzira sto globalna varijabla nije niz, opet pristupamo njome kao nizu
od jednog clana

petlja:
       CMP(edi, ebx)
       JGE(izlaz)
       MOV(eax, sum)
       ADD(eax, DWORD(niz[edi*4]))//uvecavamo sumu
       MOV(sum, eax)
       INC(edi) //i++
       JMP(petlja)


izlaz:
       MOV(eax, sum)
       PUTI(eax)

       POP(ebp)
       RET(0)
       ENDP
```

Zadatak 2. Napišite prijevod u asemblerski jezik sljedećeg C programa:

```
int GetMultiplied(int x, int y);
/* vraća vrijednost: argumenta x pomnožen argumentom y*/
{
        return x*y;
}

/*globalne varijable*/
int a = 5;
int b = 5;

void main()
{
  printf("%d", GetMultiplied (a, b));
}
```

```
VAR a DD(1) = {D_(5)};
VAR b DD(1) = {D_(5)};

//argumenti (opet esp koristimo)
#define x DWORD(M_[esp+4])
#define y DWORD(M_[esp+8])

PROC(GetMultiplied) {

       MOV(eax, x)
       ADD(eax, y)
       RET(0)
       ENDP

}


PROC (MAIN)
       PUSH(ebp)
       MOV(ebp, esp)

       //prenosimo vrijednosti globalnih varijabli
       MOV(eax, DWORD(b))
       PUSH(eax)
       MOV(eax, DWORD(a))
       PUSH(eax)

       CALL(GetMultiplied)
       ADD(esp, 8) //ponistavamo argumente


       PUTI(eax) //ispisujemo rezultat funkcije
       POP(ebp)
       RET(0)
       ENDP
```