Programação em Assembly MIPS II

1. Escreva o código em assembly MIPS correspondente ao programa em C que se segue, preservando a sua estrutura funcional (soma de uma sequência de inteiros positivos).

```
int main() {
   int i, upTo, sum;
   scanf("%d", &upTo);
   sum = 0;
   for (i = 0; i < upTo ; i++)
      sum += i * i;
   printf("The result is %d\n", sum);
   return 0;
}
```

2. Escreva o código em assembly MIPS correspondente ao programa em C que se segue, preservando a sua estrutura funcional (cálculo do quadrado de inteiros positivos).

```
int squares[100];

int main() {
   int i, upTo;
   scanf("%d", &upTo);
   for (i = 0; i < upTo ; i++)
      squares[i] = i * i;
   return 0;
}
```

3. Escreva o código em assembly MIPS correspondente ao programa em C que se segue, preservando a sua estrutura funcional (soma de uma sequência de quadrados de inteiros positivos utilizando funções auxiliares).

```
int squares[100];

void storeValues(int n) {
   int i;
   for (i = 0; i < n; i++)
      squares[i] = i * i;
   return;
}

int computeSum(int n) {
   int i, sum;
   sum = 0;
   for (i = 0; i < n; i++)
      sum += squares[i];
   return sum;
}

int main() {
   int upTo;
   scanf("%d", &upTo);
   storeValues(upTo);
   printf("The result is %d\n", computeSum(upTo));
   return 0;
}
```

4. Escreva o código em assembly MIPS correspondente ao programa em C que se segue, preservando a sua estrutura funcional (cálculo da frequência de cada letra numa frase).

```
char text[] = "This is the string to be used to generate the histogram";
int histogram[26] = {0};

void computeHistogram() {
   int i = 0;
```

```
   while (text[i] !=  '\0') {
      if (text[i] >= 'A' && text[i] <= 'Z')
         histogram[text[i] - 65]++;
      if (text[i] >= 'a' && text[i] <= 'z')
         histogram[text[i] - 97]++;
      i++;
   }
   return;
}

void printHistogram() {
   int i;
   for (i = 0; i < 26 ; i++)
      printf("%c -> %d\n", i + 97, histogram[i]);
   return;
}

int main() {
   computeHistogram();
   printHistogram();
   return 0;
}
```

5. (saber mais) Escreva o código em assembly MIPS correspondente ao programa em C que se segue, preservando a sua estrutura funcional (ordenação de um vector de inteiros pelo método da bolha).

```
int size = 10;
int array[] = {32, 6, 51, 63, 22, 29, 91, 39, 66, 47};

void bsort(int v[], int n) {
   int i, j;
   for (j = 0 ; j < n ; j++) {
      for (i = 0 ; i < n - 1 ; i++) {
         if (v[i+1] < v[i]) {
            int tmp;
            tmp = v[i];
            v[i] = v[i+1];
            v[i+1] = tmp;
         }
      }
   }
   return;
}

int main() {
   int i;
   bsort(array,size);
   for (i = 0 ; i < size ; i++)
      printf("%d ", array[i]);
   printf("\n");
   return 0;
}
```