c)

```
typedef void (*SSAMapFunction)(int index, const char *str, void *auxData)
void SSAMap(sparsestringarray *ssa, SSAMapFunction mapfn, void *auxData)
{
   int index = 0;
   for (int i = 0; i < ssa->numGroups; i++) {
      group *grp = &ssa->groups[i];
      int groupSize = ssa->groupSize;
      if (i == ssa->numGroups - 1 && ssa->arrayLength % ssa->groupSize > 0)
         groupSize = ssa->arrayLength % ssa->groupSize;

      int indexOfNonEmptyString = 0;
      for (int j = 0; j < groupSize; j++) {
         const char *str = "";
         if (grp->bitmap[j]) {
            str = *(char **) VectorNth(&grp->strings, indexOfNonEmptyString);
            indexOfNonEmptyString++;
         }
         mapfn(index, str, auxData);
         index++;
      }
   }
}
```

Solution 2: Serializing Lists of Packed Character Nodes

```
int *serializeList(const void *list)
{
   int *serialization = malloc(sizeof(int));
   int serializationLength = sizeof(int);
   const void **curr = (const void **) list;
   int numNodes = 0;

   while (curr != NULL) {
      const char *str = (const char *)(curr + 1);
      serialization = realloc(serialization,
                              serializationLength + strlen(str) + 1);
      strcpy((char *) serialization + serializationLength, str);
      serializationLength += strlen(str) + 1;
      curr = (const void **) *curr;
      numNodes++;
   }

   *serialization = numNodes;
   return serialization;
}
```