```
#include<sstream>
#include<iostream>
#include<string.h>
#include<string>
#include<fstream>
#include<string>
#include<cstdlib>

using namespace std;
bool occupied_pages[30];

int s_to_i(string operand)
  {

    if(operand[0]>='0' && operand[0]<='9' && operand[1]>='0' && operand[1]<='9')
      return ((int)operand[0]-48)*10+((int)operand[1]-48);
    return -1;
  }

class memory
{
private:
  char mem[300][4];
  char ch;
  int page_table_ptr;

public:
  void reset()
  {

    memset(mem,'$',sizeof(char)*300*4);
    memset(occupied_pages,false,sizeof(bool)*30);
    page_table_ptr=rand()%30;
    printf("%d\n",page_table_ptr);
    occupied_pages[page_table_ptr]=true;
    page_table_ptr*=10;
    printf("%d\n",page_table_ptr);
  }
  string get_mem(int  pos)
  {

    string temp="";
    for(int i=0;i<4;i++)
      temp+=mem[pos][i];
    //cout<<"String  "<<temp<<endl;
    return temp;
  }
  void set_mem(string  s, int pos)
  {
    for(int i=0;i<4;i++)
      mem[pos][i]=s[i];
  }
  int get_page_table_ptr()
  {
    return page_table_ptr;
  }
  int allocate_page()
```

```
  {
    int page_no=rand()%30;
    while(occupied_pages[page_no]==true)
      page_no=rand()%30;
    occupied_pages[page_no]=true;
    cout<<"Page no: "<<page_no<<endl;
    return page_no;
  }
  void set_page_table(int row_num,int page_no)
  {
    ostringstream temp;
    temp << page_no;
    string table_entry;
    if(page_no<10)
      table_entry="**0"+temp.str();
    else
      table_entry="**"+temp.str();
    set_mem(table_entry,page_table_ptr+row_num);
  }
  // void store_card(string s,int mem_cnt)
  // {
  //   string word="";
  //   int page_no=allocate_page();
  //   printf("%d\n",page_no);
  //   set_page_table(mem_cnt,  page_no);
  //   page_no*=10;
  //   for(int i=0;i<s.length();i+=4)
  //   {
  //     for(int j=0;j<4;j++)
  //     {
  //       word+=s[i+j];
  //     }
  //     set_mem(word,page_no);
  //     page_no++;
  //     //cout<<"Word: "<<word<<endl;
  //     word="";
  //   }
  // }
  void print_mem()
  {
    for(int i=0;i<300;i++)
    {
      string temp;
      cout<<"Data at mem location"<<"["<<i<<"]";
      for(int j=0;j<4;j++)
      {
        cout<<mem[i][j];
      }
      cout<<"\n";

    }

  }
}m_obj;

class cpu
{
```

```
public:
   int address_tranlation(int virtual_add)
   {
      int page=m_obj.get_page_table_ptr()+(virtual_add/10);
      string value_page=m_obj.get_mem(page);
      value_page=value_page.substr(2,2);
      return (s_to_i(value_page)*10+(virtual_add%10));
   }
}exe;


void GD(string s,int loc){
   int pos;
   int page_no=m_obj.allocate_page();
   m_obj.set_page_table((loc)/10,page_no);
   pos=exe.address_tranlation(loc);
   pos=(pos/10)*10;
   int len=s.length(),start=0,i;
   string s1;
   for(i=pos;start<len;i++)
   {
      if((len-start)<4)
         s1=s.substr(start,(len-start));
      else
         s1=s.substr(start,4);
      start+=4;
      m_obj.set_mem(s1,i);
   }
}



int main()
{
   std::string s;
   m_obj.reset();
   std::cout << "Enter the data: ";
   std::getline(std::cin, s);
   int loc;
   std::cout << "Enter the location:";
   std::cin >> loc;
   GD(s,loc);
   m_obj.print_mem();

   return 0;
}
```