```
// ipt.h
//      Template for what an inverted page table (ipt) entry will consist of.

#ifndef IPT_H
#define IPT_H

#include "copyright.h"
#include "openfile.h"
#include "syscall.h"
#include "machine.h"

#define IPT_HASH_TABLE_SIZE 67 // this line and the following are used to
#define PRIMESIZE 19           // enhance the performance of the hash table

#define OLD_ENOUGH 5           // "old-enough" change clock algorithm
#define DIRTY_ALLOWANCE 1      // let dirty children stay in a bit longer


//----------------------------------------------------------------------
// IptEntry
//      These classes are chained off of the IptHashTable.  They allow a
// process to make a fast connection between a virtual and physical
// address (usu. to enter into the TLB).
//----------------------------------------------------------------------

class IptEntry {
 public:
  IptEntry(int vpnArg, int phyPageArg, IptEntry *prevIptArg);
  ~IptEntry(void);

  SpaceId pid;
  unsigned int vPage;           // virtual page num
  unsigned int phyPage;         // physical page num
  IptEntry *prev;           // previous pointer
  IptEntry *next;           // next pointer
};

//----------------------------------------------------------------------
// IptHashTable
//      Used to access IptEntries (which are chained off of this class).
// See above for constants used to make lookup quick.
//----------------------------------------------------------------------

class IptHashTable {
 public:
  IptHashTable(void);
  ~IptHashTable(void);

  IptEntry *entries;            // chained entries
};

//----------------------------------------------------------------------
// MemoryTable
//      Used by the clock algorithm to choose physical pages when things
// need to be swapped in.  Lots of fields are used for efficiency.
// One class per page frame.
//----------------------------------------------------------------------

class MemoryTable {
 public:
  MemoryTable(void);
  ~MemoryTable(void);

  bool valid;               // if frame is valid (being used)
  SpaceId pid;              // pid of frame owner
  int vPage;                // corresponding virtual page
  IptEntry *corrIptPtr;         // corresponding IptPtr
  bool dirty;               // if needs to be saved
  int TLBentry;             // corresponding TLB entry
  int clockCounter;         // used to see how much it's being used
  OpenFile *swapPtr;            // file to swap to
};

//----------------------------------------------------------------------
// hashIpt
//      Function to hash into IptHashTable.  Take a virtual page number
// and process Id.  Returns an IptEntry.
//----------------------------------------------------------------------

IptEntry *hashIPT(unsigned int vpn, SpaceId id);

#endif // IPT_H

```