Help

```
#ifndef  _VAR_H
#define _VAR_H

#include "
href../common/enums_h_src.pdfenums.h"


extern int Fprintf(int user, const char *s, ...);
extern int FprintfVar(int user, const char s[], const VAR *x);
extern int FScanVar(char **InputFile, Planning *pt_plan, int user, VAR *x);

extern int Valid(int user, int status, char *helpfile);
extern int FGetParVar(char **InputFile, Planning *pt_plan, int user, VAR *x);

extern int InitVar(void);
extern void ExitVar(void);
extern int PrintVarRec(const Planning *pt_plan, int user, const VAR *, int isrec
extern int PrintVar(const Planning *pt_plan, int user, const VAR *);
extern int ScanVar(Planning *pt_plan, int user, VAR *);
extern int ChkVar(const Planning *pt_plan, VAR *x);
extern int ChkVar1(const Planning *pt_plan, VAR *x, int tag);
extern int ChkVarLevel(const Planning *pt_plan, VAR *x);
extern int GetParVar(Planning *pt_plan, int user, VAR *x);
extern int ShowParVar(const Planning *pt_plan, int user, const VAR *x);
extern int ChkParVar(Planning *pt_plan, VAR *x);
extern int LowerVar(int user, VAR *x, VAR *y);
extern void CopyVar(VAR *srce, VAR *dest);
extern int CheckIterationValue(char **InputFile, Planning *pt_plan, int user, VA

extern void ResetPlanning(Planning *pt_plan);
extern void ShowPlanning(int user, const Planning *pt_plan);
extern void ShrinkPlanning(int index, Planning *pt_plan);
extern int ChkStepNumber(int user, Iterator *pt_iterator, int step);
extern void NextValue(int count, Iterator *pt_iterator);
extern void premia_Vtype_info(VAR *x, char **format, char **error_msg, int *type
extern PremiaEnumMember *lookup_premia_enum(const VAR *x, int key);
extern PremiaEnumMember *lookup_premia_enum_with_index(const VAR *x, int key, in
extern VAR *lookup_premia_enum_par(const VAR *x, int key);
```

```
extern int ShowParVarTestRes(Planning *pt_plan, int user, VAR *x);

#ifndef _WIN32
extern int Spawnlp(int mode, const char *cmdname, const char *arg0, const char *
#endif

extern int charPtr_to_PnlVect(PnlVect *x, const char *s);

extern int premia_clone_vars(VAR **res, int flag, const VAR *vars, int n);

extern void free_premia_model(Model *);
extern void free_premia_option(Option *);
extern void free_premia_method(PricingMethod *);
extern void free_premia_var(VAR *x);

#endif
```